Webots

1. Download

Download Webots from Cyberbotics (free version):
https://cyberbotics.com/

2. Installation

For the installation of Webots you should follow the instructions on https://cyberbotics.com/doc/guide/installation-procedure?tab-language=python where you'll find the procedure for Windows, Linux and MacOS.

3. Setting up Webots with Python

The assignments for the class will be coded using Python. Follow the instructions on https://cyberbotics.com/doc/guide/using-python?tab-language=c, if you have not yet downloaded Python in your computer.

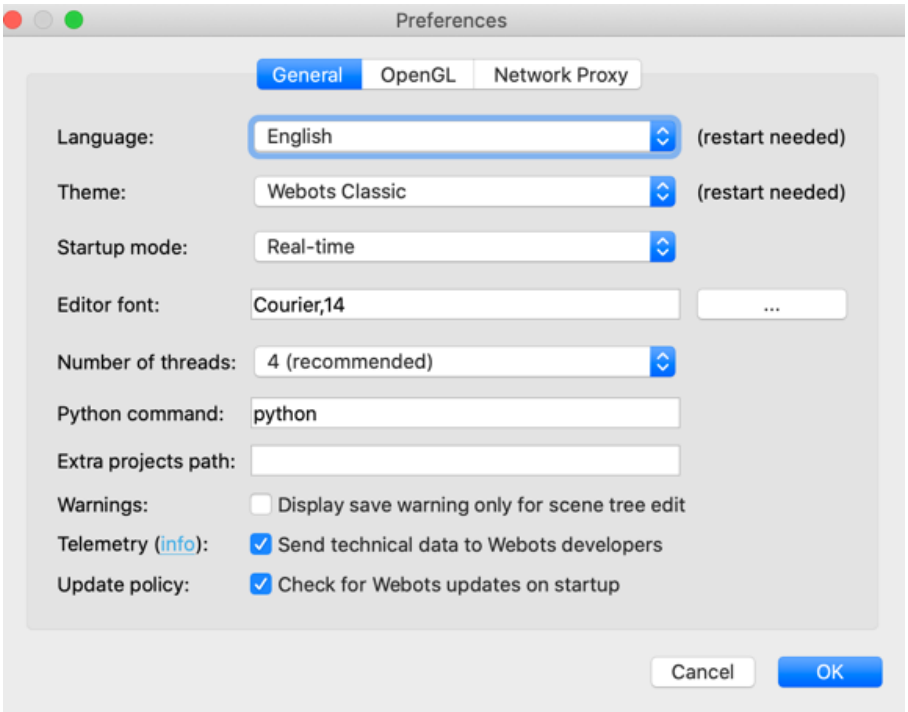
Notes:

- Webots requires Python version 3.7 or later, www.python.org.
- Set your current PATH to your latest Python version.

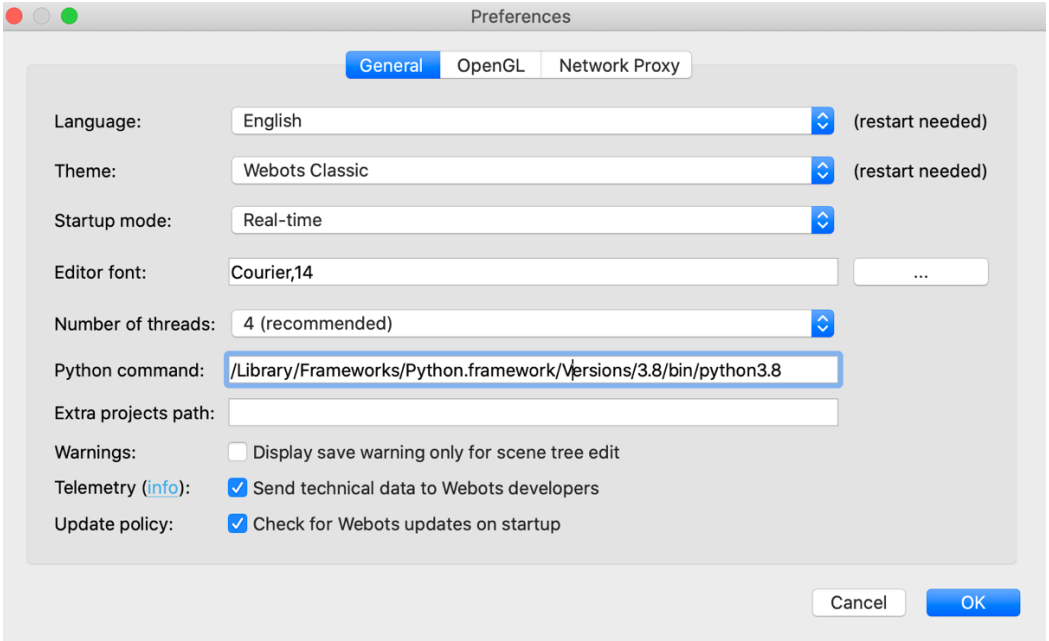
MacOS Python update

Download: https://www.python.org/downloads/mac-osx/

Original Webots "python" command configuration:



Updated path "/Library/Frameworks/Python.framework/Versions/3.8/bin/python3.8":



4. Webots Tutorial

Webots includes a main tutorial that you should read and follow:

https://cyberbotics.com/doc/guide/tutorials

5. Structure of a Webots project

Although "empty" worlds will be provided for each lab, containing the environment and robot, you should learn how to create a new world in a project.

- A world is made up of nodes organized in a tree structure.
- A world is saved as a .wbt file stored in a Webots project.
- The project also contains the robot controller programs which define the behavior of the robots.
- Controllers are associated with robots via the controller fields of the Robot node.

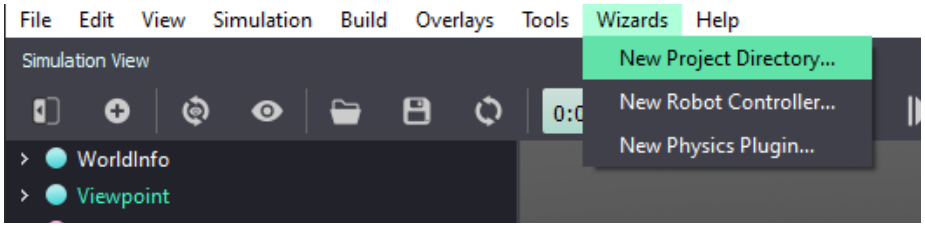
5.1. Create a New World

A **World** is a file containing information like where the objects are, what they look like, how they interact with each other, what is the color of the sky, and other aspects such as, gravity, friction, masses of the objects, etc. The world defines the initial state of a simulation. The different objects are called **Nodes** and are organized hierarchically in a **Scene Tree**. Therefore, a node may contain sub-nodes. A world is stored in a file having the .wbt extension. The file format is derived from the **VRML97** language, and is human readable. The world files must be stored directly in a directory called worlds.

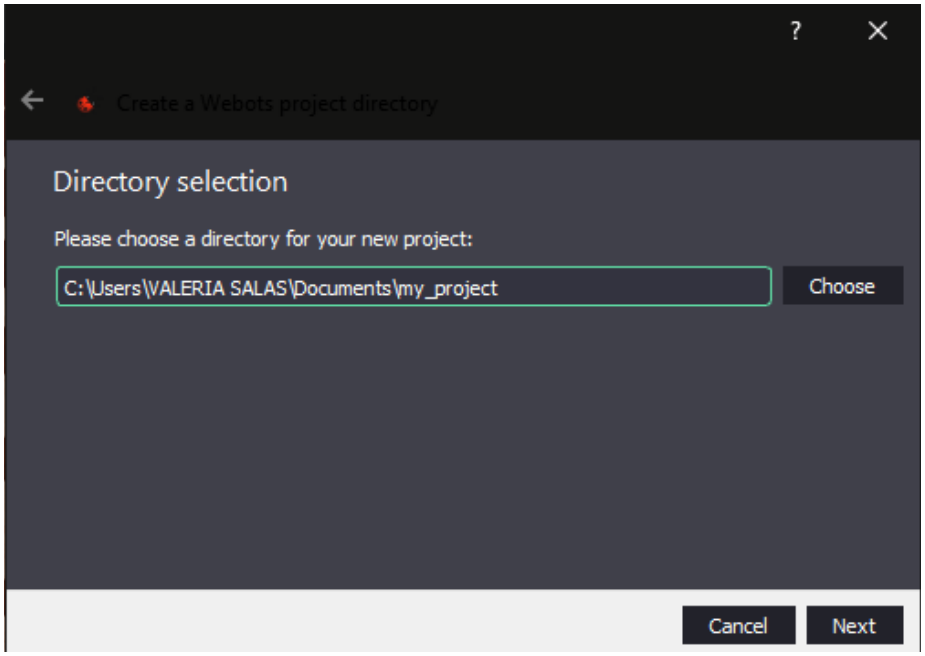
To create a new World:

1. Pause the current simulation by clicking on the Pause button  of the 3D view. The simulation is paused if the virtual time counter on the main toolbar is stopped.

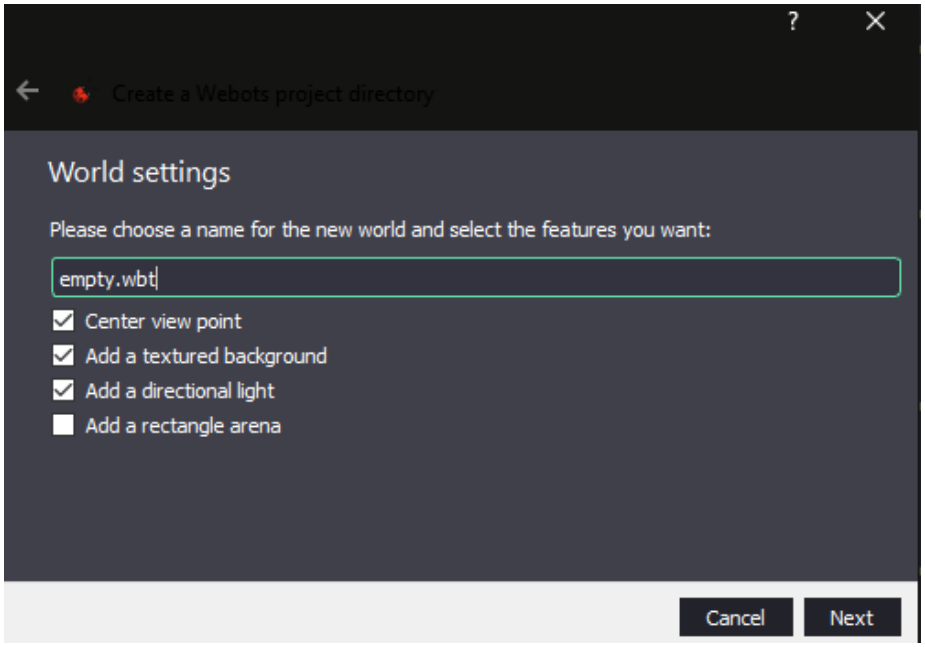
2. Create a new project from the Wizards menu by selecting the New Project Directory...



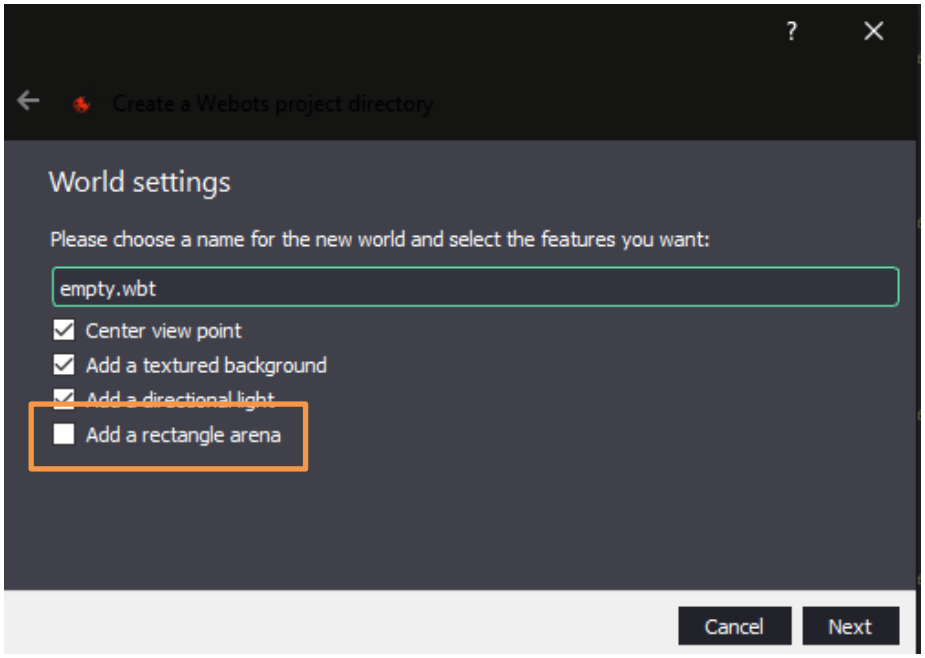
3. Change the directory of the project



4. Change the name of the world file



5. Click all the boxes, including the "Add a rectangle arena" which is not clicked by default. You will be able to edit this arena once you start working on your world.



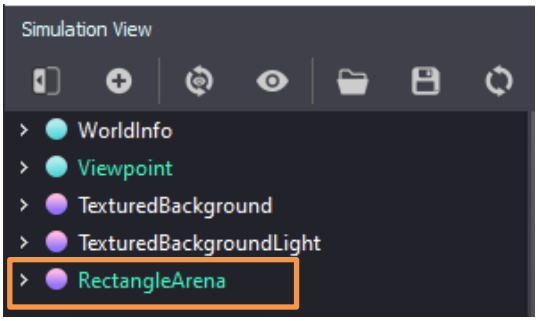
5.2. Add a robot (“E-puck”) to the world

The E-puck is the robot that you will be using for all of the assignments.

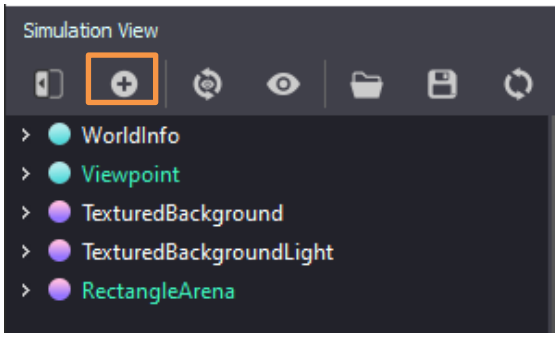
This robot is located in the folder "WEBOTS_HOME/projects/robots/gtronic/e-puck/protos/E-puck.proto".

To add an E-puck Robot in you world:

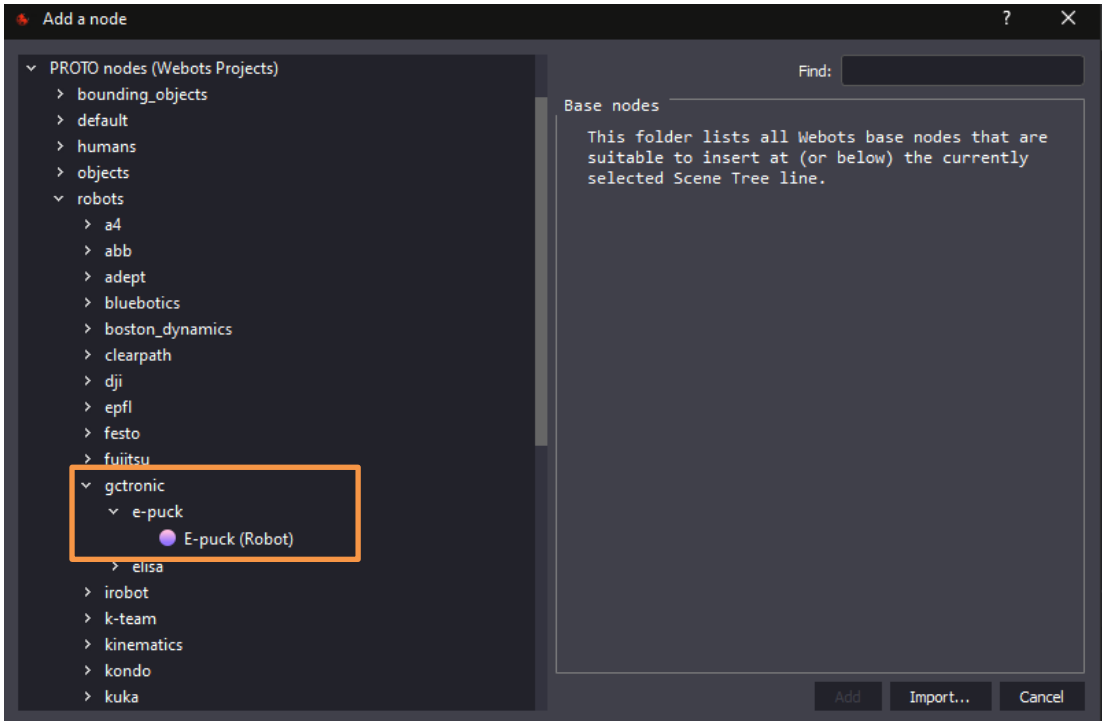
1. Select the last node of the scene tree view.



2. Click on the Add button  at the top of the scene tree view.



3. In the dialog box, choose PROTO nodes (Webots Projects) / robots / gctronic / e-puck / E-puck (Robot).



An E-puck robot should appear in the middle of the arena.

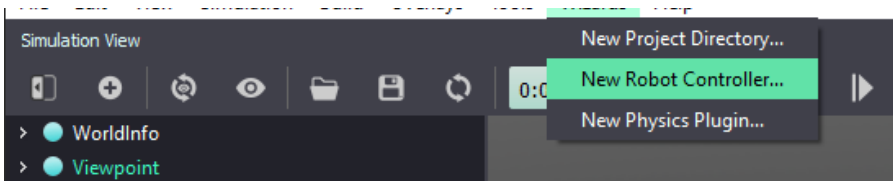
5.3. Create a new Controller

A **controller** is a program that defines the behavior of a robot. When you add a new robot to your project, it will have a default controller linked to it.

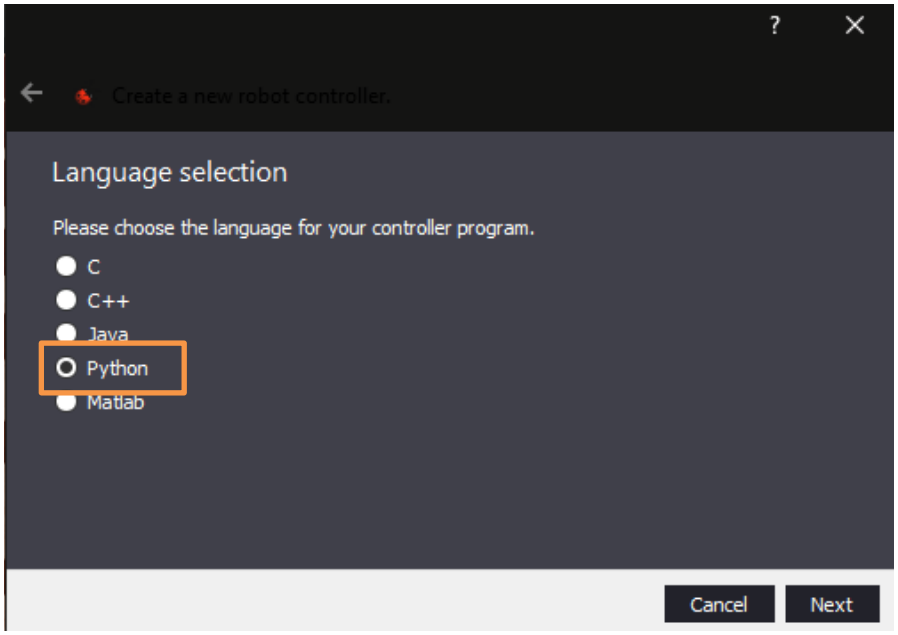
The controller field of a Robot node specifies which controller is currently associated to the robot. Note that the same controller can be used by several robots, but a robot can only use one controller at a time.

To create a new controller:

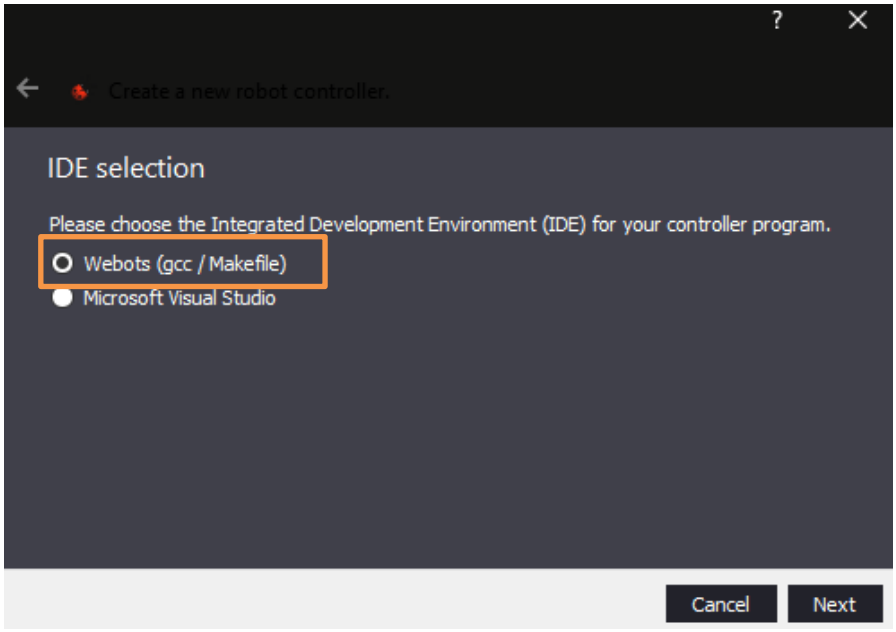
1. Go to the Wizards / New Robot Controller... menu, and then click next.



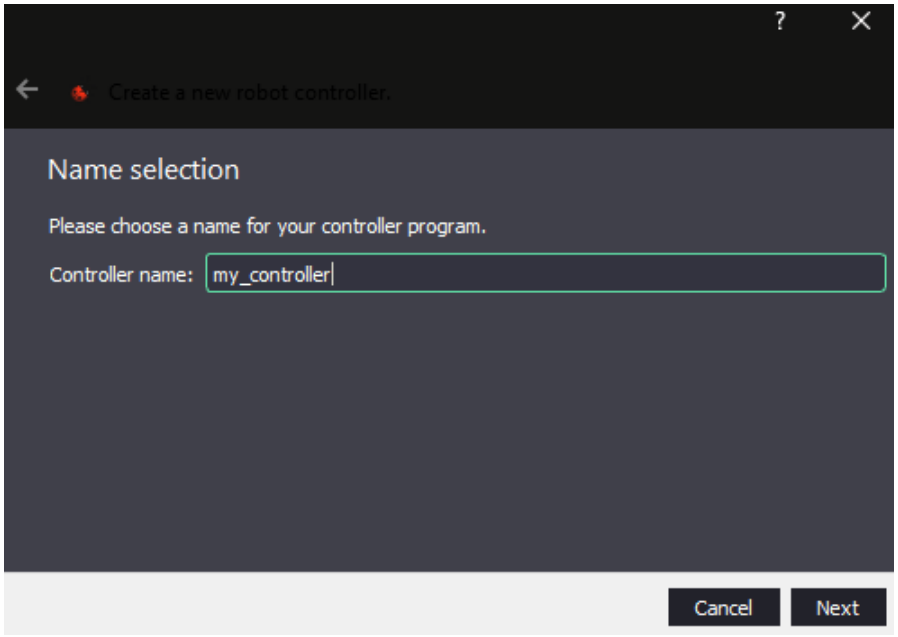
2. Select Python as the language of your controller.



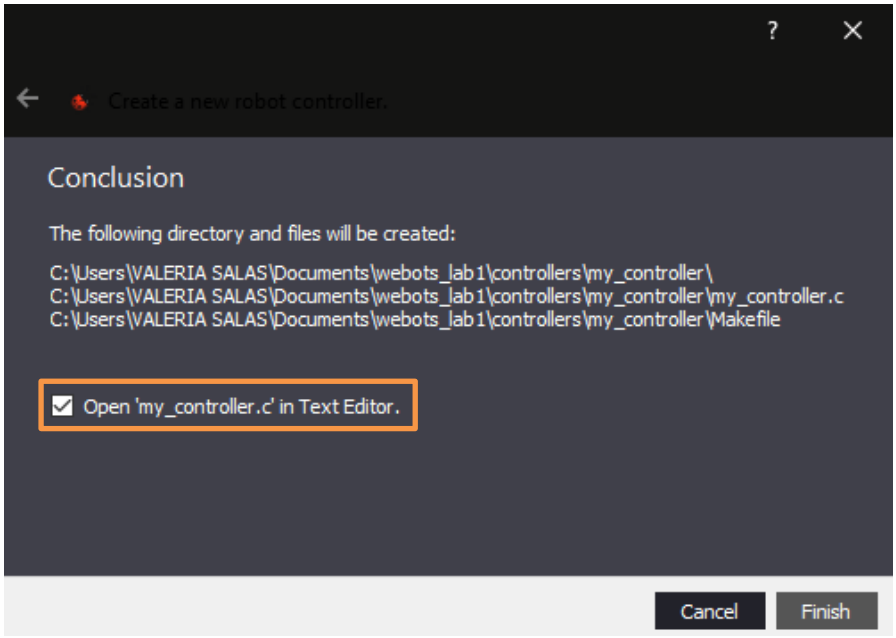
3. Select Webots as your IDE for programming



4. Change the name of your new controller:



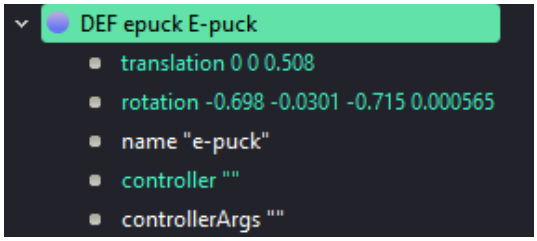
5. Select the option offering you to open the source file in the text editor and select 'Finish'.



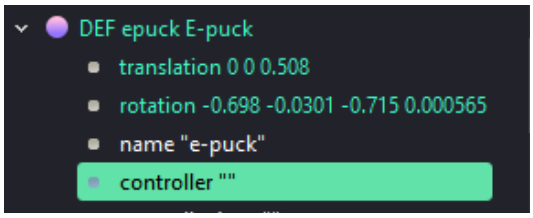
After you follow these steps, the code for your controller will appear in a text editor on the right of the screen.

After you create your controller, you will have to link it with your E-puck:

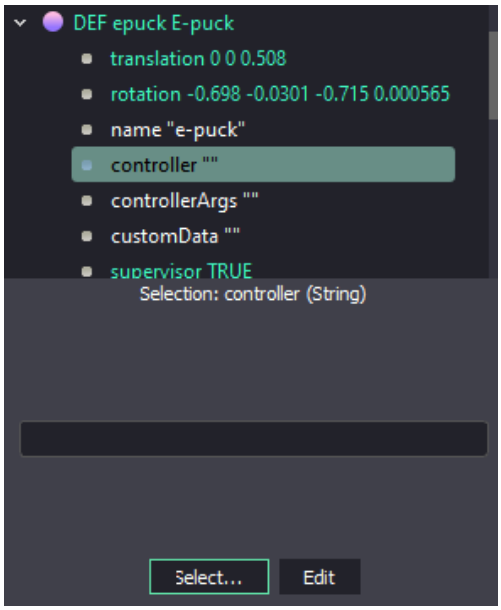
1. Double click in your ‘E-puck’ node in the Scene Tree view. This will open the node and display its fields.

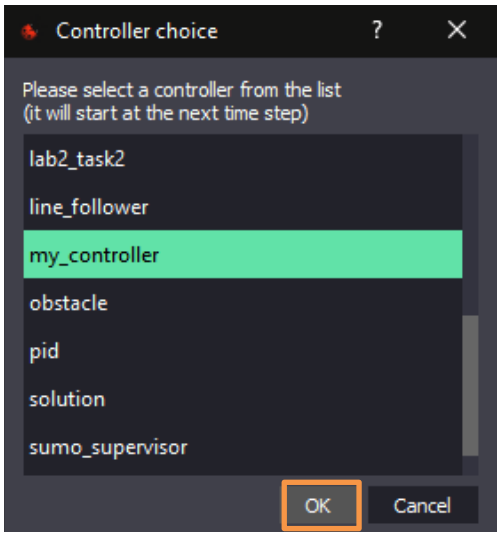


2. Double click on the ‘controller’ field from E-puck node in the Scene Tree View

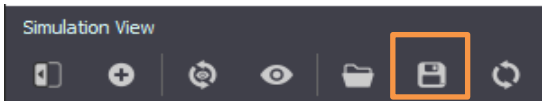


3. Use the field editor at the bottom of the Scene Tree view: press the Select... button and then select the controller you previously created or from any controller you may have on the list and select ‘OK’.





4. Once the controller is linked to the robot, save the world.



You can follow these steps to create as many controllers as you want or change between controllers. Note that Webots allows a single controller per folder and only one controller can be linked to your robot at once.

Now you can modify your controller. Note that you will have to import some classes to use the different devices of your E-puck, in our case we import the class **Motor** to be able to use the motors of this robot, and the same with cameras and sensors.

This is an example of code for a two-motor robot:

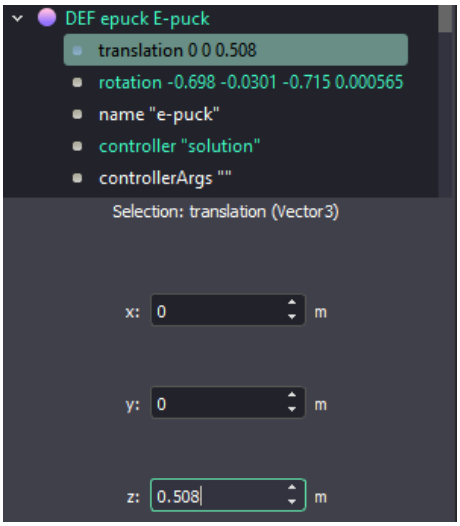
```python
from controller import Robot, Motor

TIME_STEP = 64

# create the Robot instance.
robot = Robot()

# get the motor devices
leftMotor = robot.getMotor('left wheel motor')
rightMotor = robot.getMotor('right wheel motor')
# set the target position of the motors
leftMotor.setPosition(10.0)
rightMotor.setPosition(10.0)

while robot.step(TIME_STEP) != -1:
    pass
```

With this code you are referring to the two motors of your E-puck and how you will be referring to all the devices of your robot, e.g. camera, distance sensors, LEDs, etc.

6. Running your Webots project

6.1. Setting the initial position of the robot

You can check the current position of your robot at any time in the Scene Tree View on the e-puck node:



You can use this tool to know your robot's coordinates at any time of the simulation or outside of the simulation. Note that the origin of the coordinate system (0, 0, 0) is always at the center of your arena.

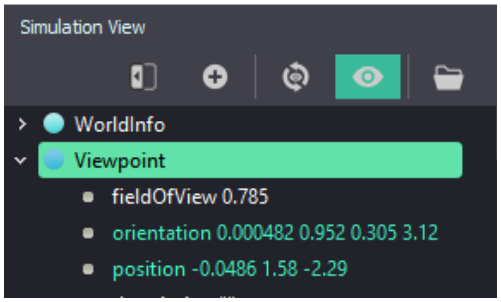
6.2. Viewing the world from different angles

6.2.1. The Viewpoint node defines a specific location in the local coordinate system from which the user may view the scene.

6.2.2. The position and orientation fields of the Viewpoint node specify absolute locations in the coordinate system. In the default position and orientation, the viewer is on the z-axis, looking down the *-z*-axis toward the origin with *+x* to the right and *+y* straight up.

6.2.3. Navigating in the 3D view by dragging the mouse pointer dynamically changes the position and the orientation fields of the Viewpoint node.

6.2.4. The fieldOfView field specifies the viewing angle in radians. A small field of view roughly corresponds to a telephoto lens; a large field of view roughly corresponds to a wide-angle lens.



6.3. Record your simulation

6.3.1. To submit your projects, you are going to record your simulation using the button to record on Webot's menu:

6.3.2. Select the default settings and then save your video in your desired location: