USI/SUPSI
IDSIA

Università della Svizzera italiana

Part 3

# Introduction to ROS

Elia Cereda, Simone Arreghini

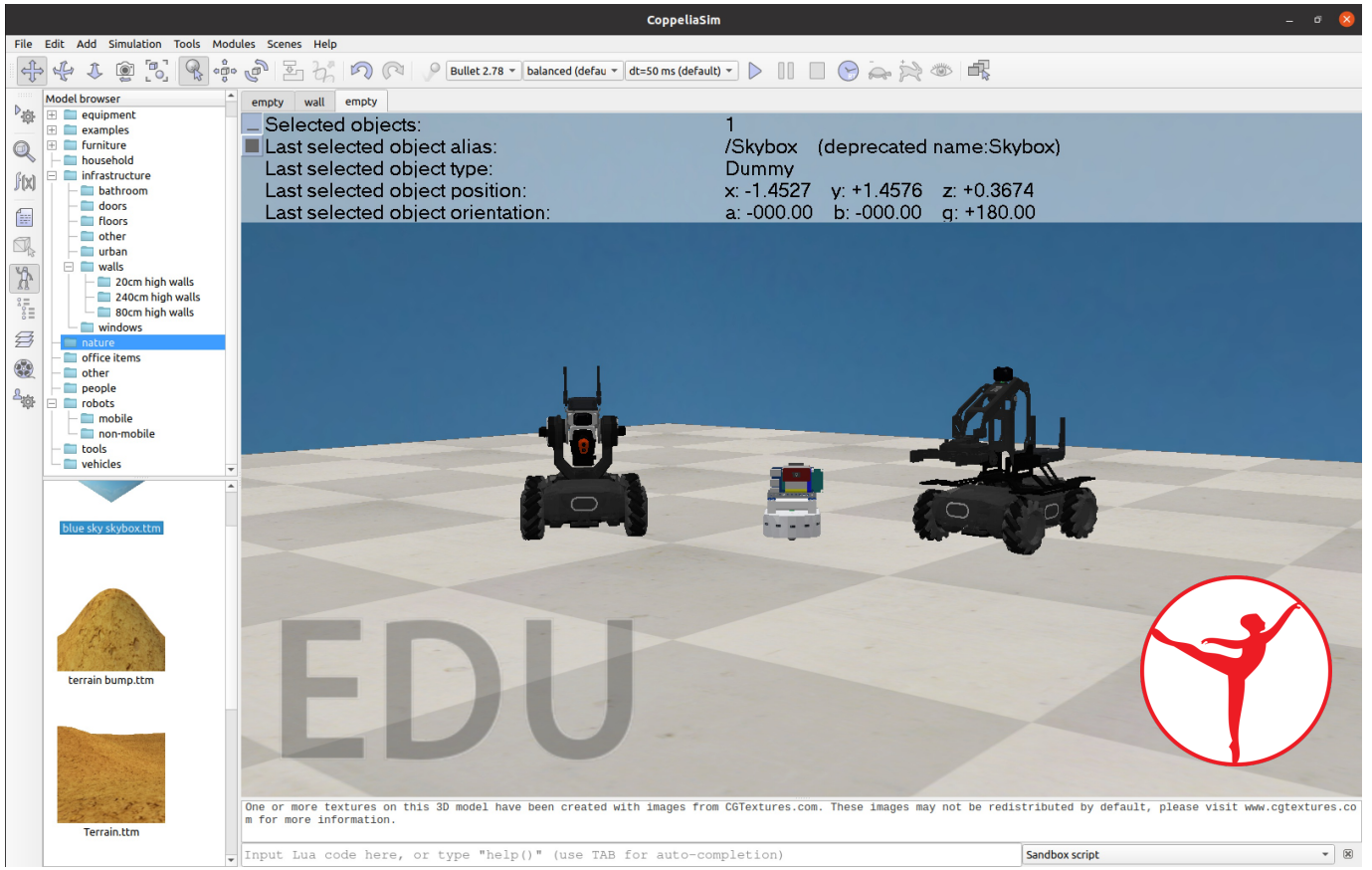
Introduction to ROS - Part 3

# CoppeliaSim

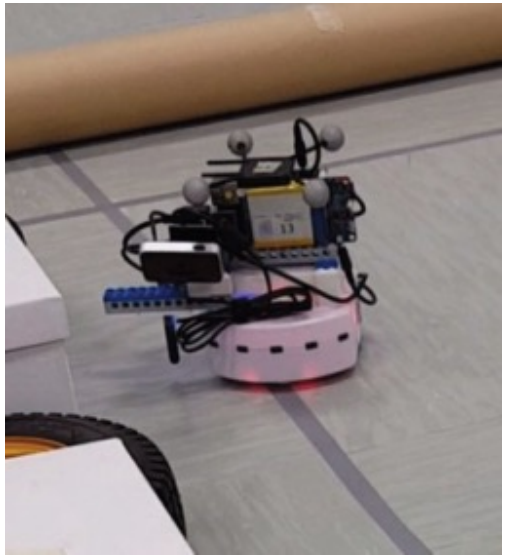
Introduction to ROS - Part 3

# CoppeliaSim

- Formerly known as V-REP, www.coppeliarobotics.com
- Robotics/Physics simulator
- 3D rigid body dynamics
- Sensor simulation & sensor design (including noise)
- Repository of robot models
- ROS2 interface
- Plugin development (Python, Lua, C++)



# We'll use the Mighty Thymio (MyT)



Base

- Length: 11.0 cm
- Width: 11.18 cm
- Height: 7.7 cm

Wheels

- Radius: 2.2 cm
- Width: 1.5 cm
- Axis: 9.35 cm
- Maximal angular speed: ~7.5 rad/s

Kinematics

- Maximal linear speed: ~ 14 cm/s
- Maximal angular speed: ~ 3 rad/s

Proximity (and ground) sensors

- Maximal range (to detect a white surface): ~ 12 cm
- Field of view: 0.3 rad

Camera

- The camera is mounted on a tilt-able joint (downward pitch is positive)
- Min pitch: -0.34 rad
- Max pitch: 1.3 rad

# But we can also simulate these two

You can use them for the final project but NOT for the second Homework

# DJI Robomaster S1

- Weights ~3.3kg
- Mecanum wheels
- Two-axis gimbal
- HD camera with wide field of view (120$^{o}$)
- Programmable LEDs
- Expansible with additional sensors
- Can be controlled via an app
- Four closed-loop motors
- Bumper hit detectors
- IR and pellet gun
- More specs here https://www.dji.com/ch/robomaster-s1/specs

# DJI Robomaster EP

- Weight ~3.3kg
- Mecanum wheels
- Controllable arm with a gripper
- HD camera with wide field of view (120º)
- Programmable LEDs
- Expansible with additional sensors
- Can be controlled via an app
- Four closed-loop motors
- Bumper hit detector
- Additional sensor adapter
- More specs here https://www.dji.com/ch/robomaster-ep-core/specs

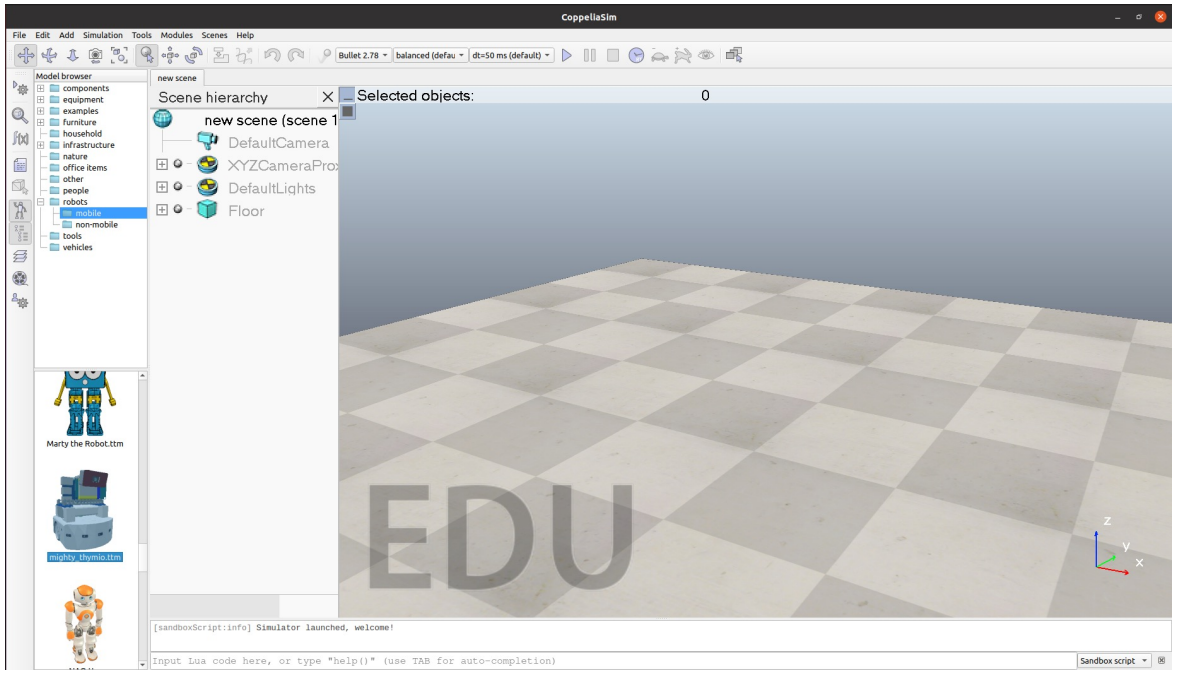
Introduction to ROS - Part 3

# Launching Coppelia

- Coppelia is already installed in the VM. Open a new terminal and start it:

  **`~/apps/CoppeliaSim_Edu.../coppeliaSim.sh`**

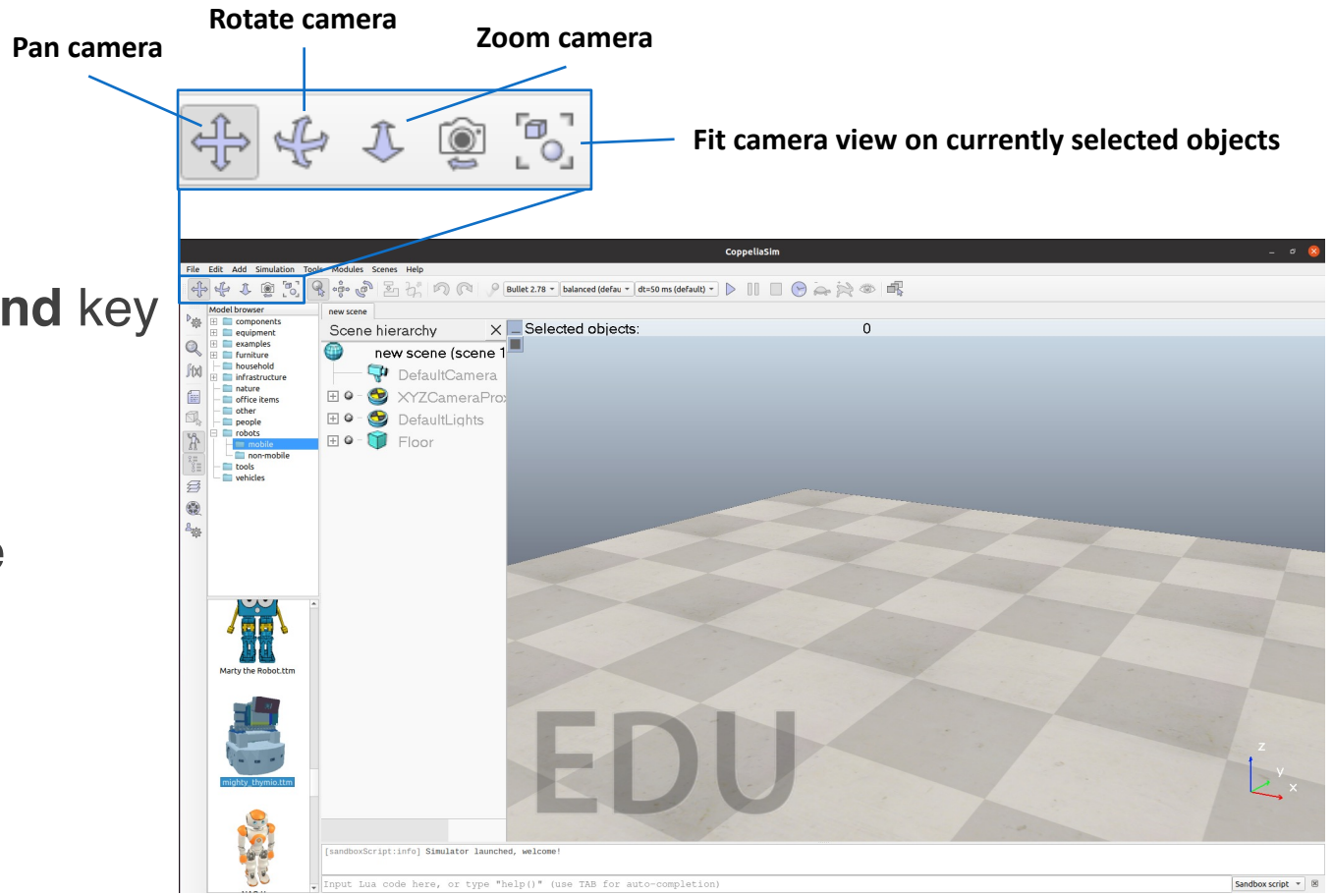
- The simulator will launch with an empty world



# Controlling the camera

- **Drag** to pan (move) the camera
- **Drag** while pressing the **Command** key to rotate the camera in place
- **Scroll** to zoom in and out

Alternatively use the buttons in the toolbar to control the camera



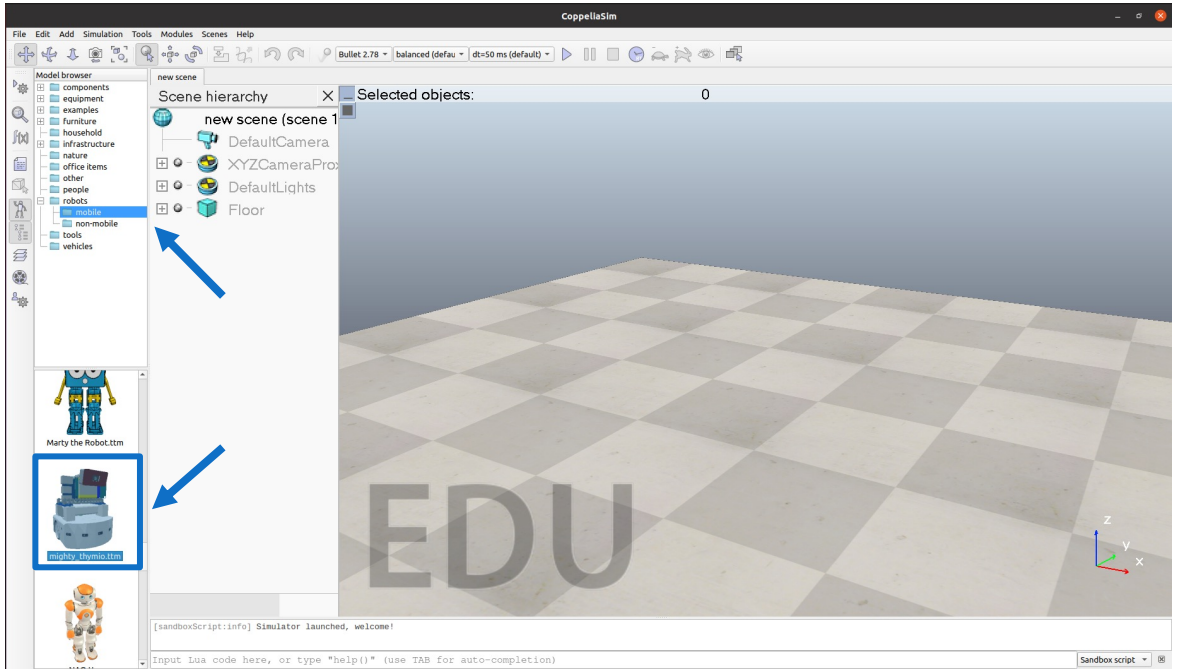
Introduction to ROS - Part 3

# Adding the first object

- You can add objects to the world from the **Model browser** on the left

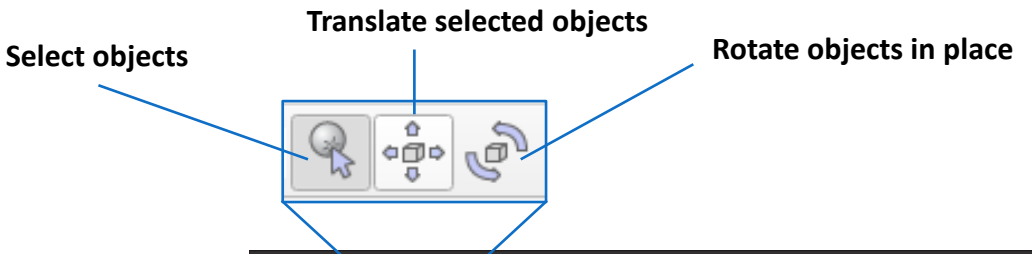
**Let's try adding a Mighty Thymio**

- Find the model in `robots` -> `mobile`
- Drag and drop it in the world

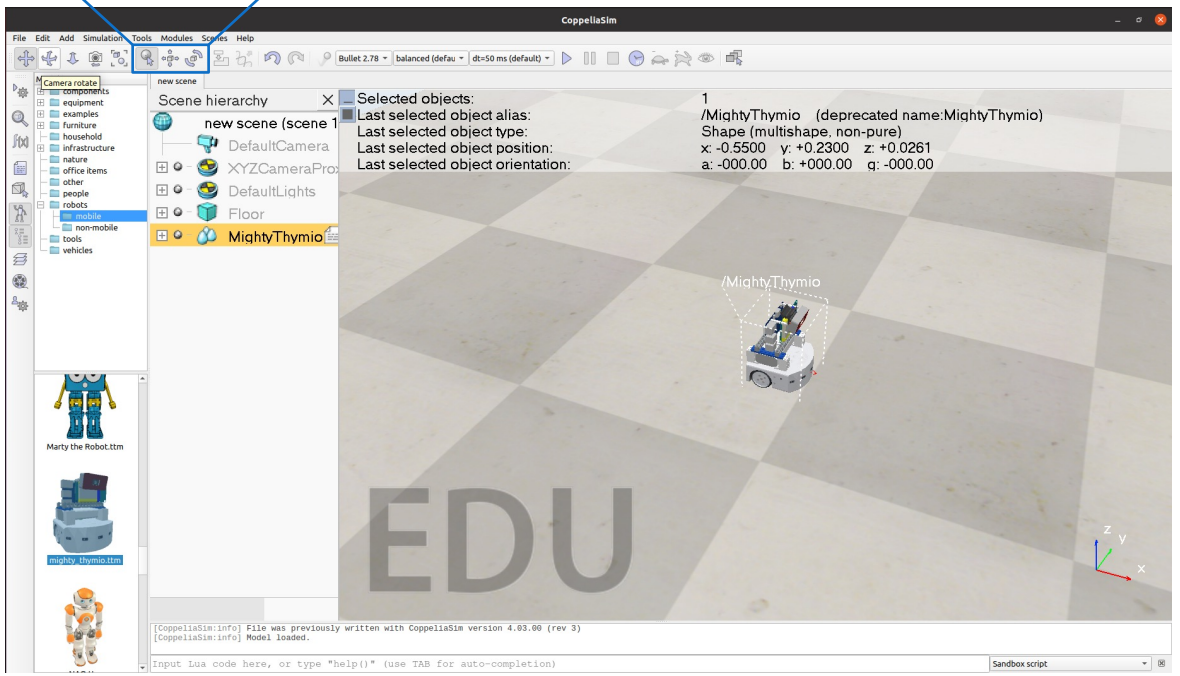


Introduction to ROS - Part 3

# Interacting with objects




- Use these tools to interact with objects when editing the world



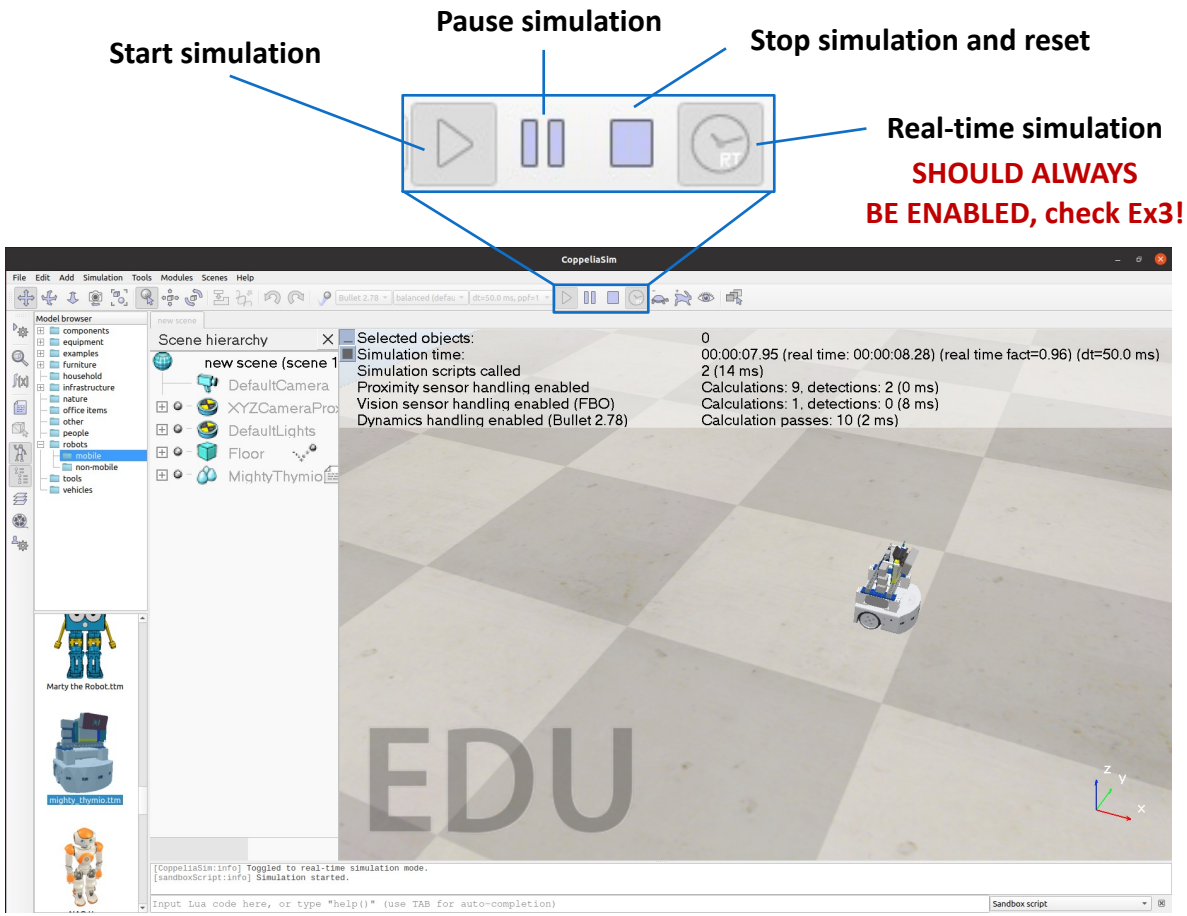
Introduction to ROS - Part 3

# Let's start the simulation

- Enable **Real-time simulation mode**

**This is important:** by default, Coppelia runs the simulation as fast as your computer can manage.

- Press the **Start** button in the toolbar, Coppelia will begin simulating the world:
  - Physics engine will become active
  - You can start sending commands to control your robot
- Press the **Stop** button to reset the world to its initial state
  - All objects will return to their initial poses

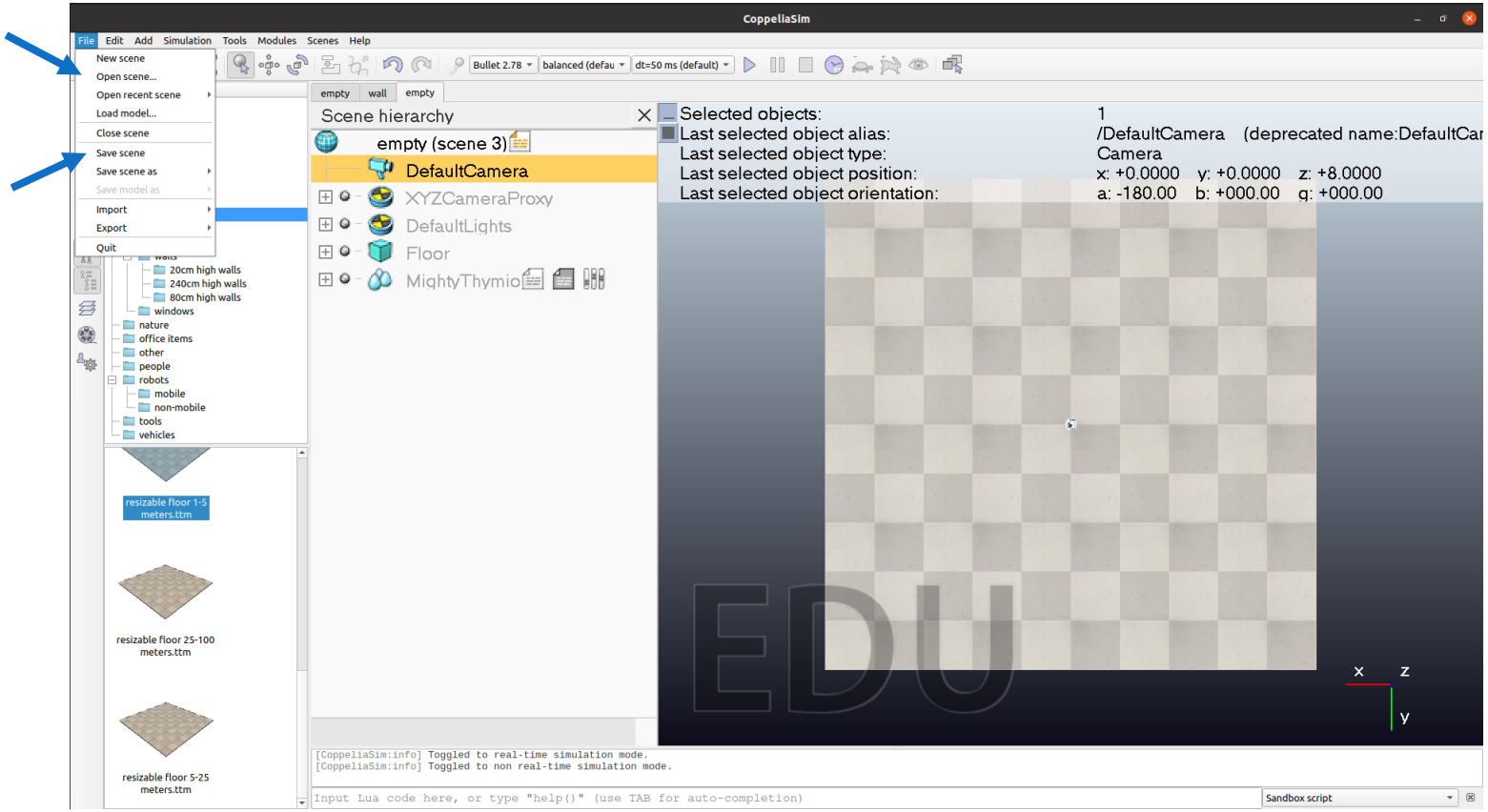


Introduction to ROS - Part 3

# Save and load scene



- You can save and load scenes from the **File** menu. You will need this for the homework!



Introduction to ROS - Part 3

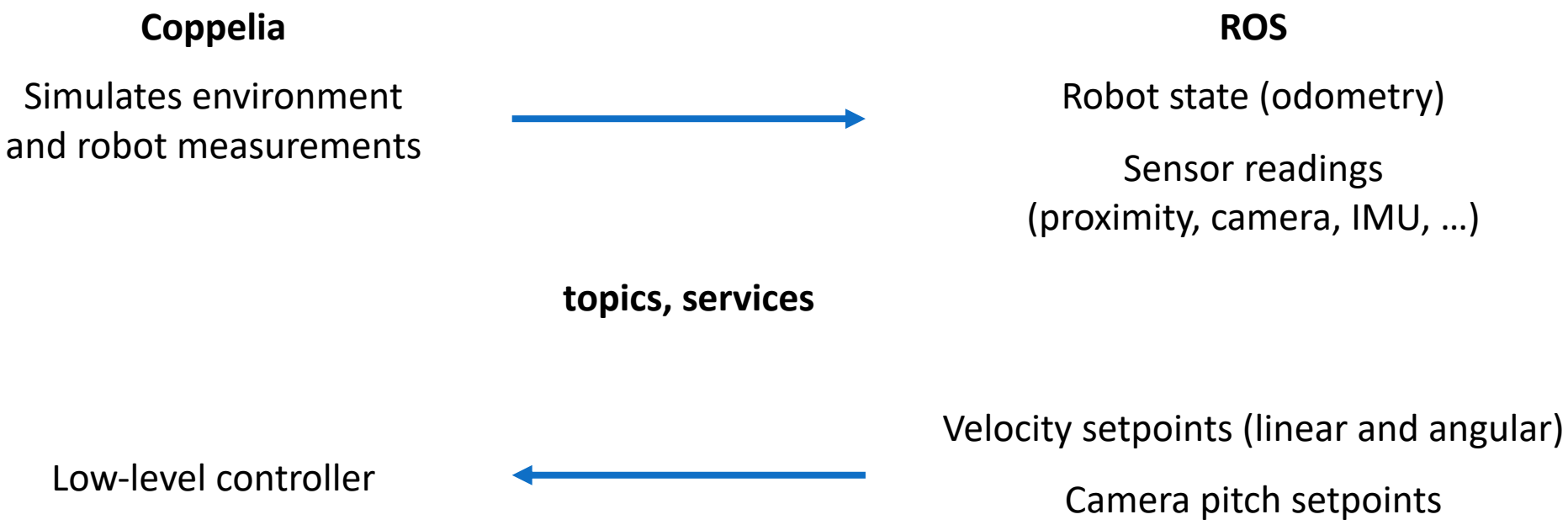
# Coppelia and ROS

- Coppelia has a modular architecture that can be extended with plugins to integrate the simulator with external systems
- *simExtROS2* is the official plugin to communicate with ROS 2, exposing the state and control interfaces of the simulated robots
- It integrates with ROS using Topics and Services (same as turtlesim)

Reference
https://www.coppeliarobotics.com/helpFiles/en/ros2Interface.htm

# Mighty Thymio, Coppelia and ROS

- We designed a MyT model in Coppelia that exposes a low-level controller interface
- Using *simExtROS2,* we interact with this interface from ROS



# Mighty Thymio Packages

The following packages **ARE ALREADY INSTALLED** on your VMs, we provide installation instructions just for your reference.

Move to your sources root with `cd ~/dev_ws/src`, then for each of the following repositories do `git clone --recursive <url> -b <branch>`

- **Aseba (low-level communication):** https://github.com/jeguzzi/ros-aseba, branch `ros2`
- **Thymio:** https://github.com/jeguzzi/ros-thymio, branch `master`
- **Mighty Thymio:** https://github.com/jeguzzi/thymioid, branch `ros2`

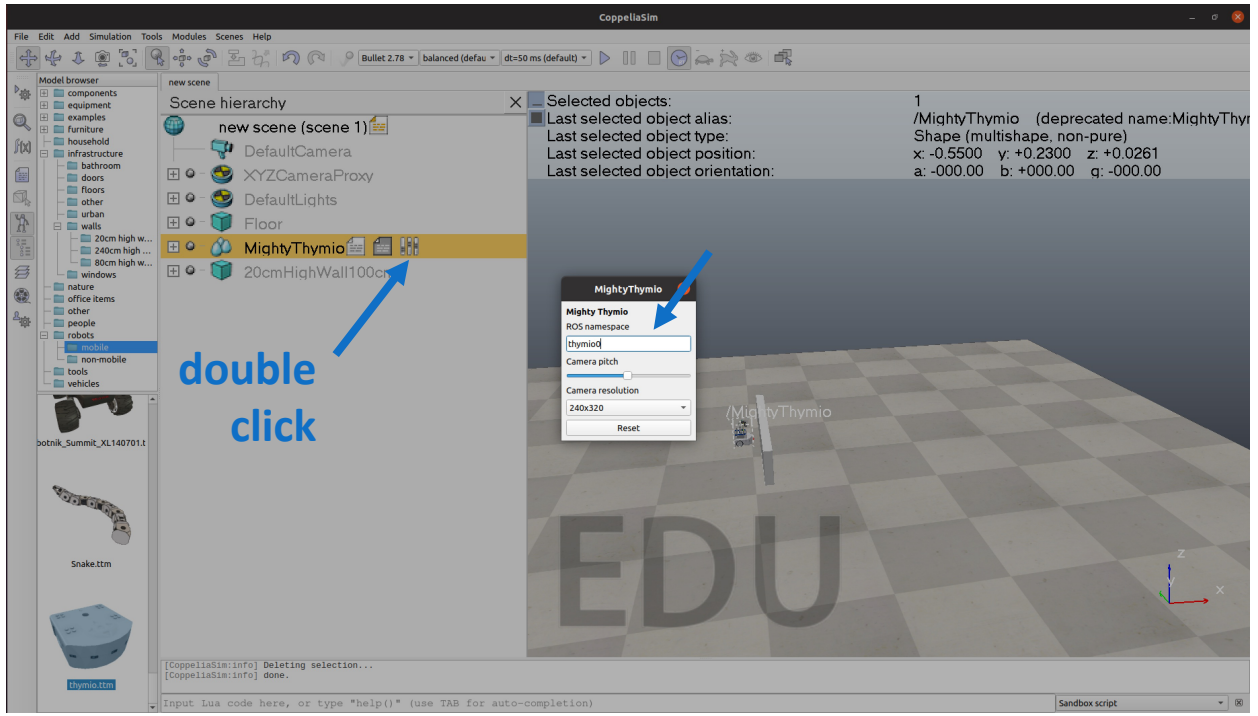
# Robomaster packages

In addition, all Robomaster packages are already installed and tested in the VM.

Spoiler: you can use the RM simulation for the final project

# Exercise 1: Simulate a Thymio

- Open Coppelia, add a Mighty Thymio to the world (use the previous slides as reference)
- **Stop the simulation**, if running, and set the **ROS namespace** in the robot's parameters



# Exercise 1: Simulate a Thymio

- In a **new terminal,** start the Mighty Thymio's ROS bridge with these commands:

  `source ~/dev_ws/install/setup.bash` (don't forget to source your workspace!)

  ```
  ros2 launch thymioid main.launch device:="tcp:host=localhost;port=33333"
  simulation:=True name:=thymio0
  ```

This will connect to the simulated Thymio and expose all the relevant data as ROS topics under the `/thymio0/*` namespace.

`ros2 launch` is used to start a set of nodes defined in a **launch** file, allowing to easily start complex systems.

- Tutorials: https://docs.ros.org/en/humble/Tutorials/Intermediate/Launch/Launch-Main.html

**Important: if you stop the simulation, you need to restart the ROS bridge!**

# Exercise 1: Simulate a Thymio

- Now `ros2 topic list` will display a lot of *topics*
- The Thymio ROS package publishes a lot of information about sensors, odometry and camera feed
- And now …

**Let's move the Thymio!**

# Exercise 1: Simulate a Thymio

- Clone the repo https://github.com/EliaCereda/thymio_example in your workspace with:

```
cd ~/dev_ws/src
git clone https://github.com/EliaCereda/thymio_example.git
```

- Install the dependencies of this package:

```
source ~/dev_ws/install/setup.bash
rosdep install -i --from-path thymio_example --rosdistro humble -y
sudo pip3 install transforms3d
```

  - **`rosdep`** is the default method for installing the dependencies of a ROS package.
    In this case, **`transforms3d`** must also be installed manually because it doesn't support **`rosdep`**.

- Build your workspace

```
cd ~/dev_ws
colcon build
```

# Exercise 1: Simulate a Thymio

- Now, open a **new terminal**, **source your workspace** and type:

```
ros2 launch thymio_example controller.launch.xml thymio_name:=thymio0
```

- *NOTE: the* **`thymio0`** *argument is passed to the node to set the robot to be controlled. Check the launch file and node script to see how it works.*
- Now in Coppelia we should see the Thymio moving forward
- To stop it, simply terminate the controller by selecting its terminal and hitting *Ctrl + C*
- *NOTE: when the simulation is stopped, nothing will happen. If you find yourself in this situation, hit the Play button in Coppelia and things will start running again.*

# rqt

- **rqt** is an essential tool for developing with ROS
- **rqt** provides different functionalities as
  - Visualizing topics (e.g., image coming from cameras)
  - Plot topics data through time (e.g., speed)
  - Configuring nodes parameters (e.g., limiting the maximum speed of a robot)
  - Controlling a robot through a simple controller called Robot
  - Steering
  - …
- All the **rqt** functionalities are implemented as dock-able GUI plugins
- Documentation: https://docs.ros.org/en/humble/Concepts/About-RQt.html

Introduction to ROS - Part 3

# Exercise 2: Thymio POV

- Ensure the simulation from the previous exercise is up and running
- Open **another terminal** and type **`rqt`**
- Click on the menu item *Plugins > Visualization > Image View:* this will show us the images from the camera of the simulated Thymio
- Select the camera topic in the top dropdown menu, **`/thymio0/camera`**

- Are you noticing anything interesting? Go to Coppelia and add some obstacles in front of the robot!

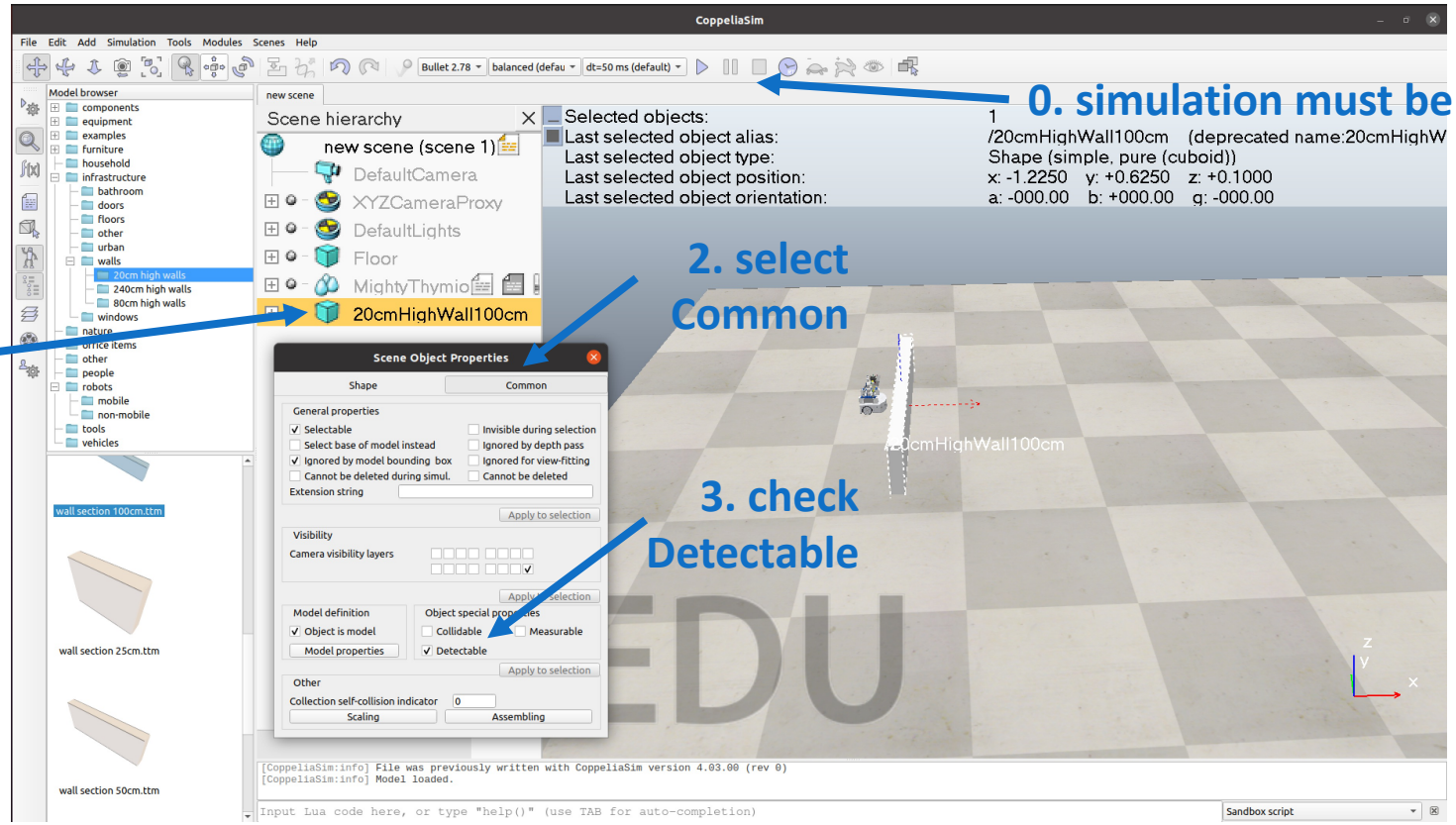
# Exercise 2: Thymio POV

- Since the Thymio is fitted with proximity sensors we can play also with those
- First open the plotting tool by clicking on the menu bar *Plugins > Visualization > Plot*
- In the text box called "Topic", type **`/thymio0/proximity/left/range`**
  - You should see a fixed line at -1
- Now try placing an obstacle very close to the left of the Thymio (< 5 cm).
  - Ex.: *infrastructure -> walls -> 20cm high walls -> wall section 100cm*

**What happens?**

# Exercise 2: Thymio POV

- You must mark objects as **detectable** to interact with the Thymio's proximity sensors



# Exercise 2: Thymio POV

- We have to do some fixes due some quirks of Humble
- in a terminal install this package

```
sudo apt install ros-humble-rqt-robot-steering
```

Say yes, and press enter if a whole terminal message is displayed asking with service should be restarted

Introduction to ROS - Part 3

# Exercise 2: Thymio POV

- Now let's try to manually control the Thymio
- First, check that **Coppelia,** the **ROS bridge,** and **rqt** are running
- Usually the Robot steering plugin is opend by clicking in the top menu bar *Plugins > Robot Tools > Robot Steering* **BUT** due to the quirk of humble we have to launch it by hand

```
ros2 run rqt_robot_steering rqt_robot_steering --force-discover
```

- We see two sliders, representing the linear and angular velocities
- At the top of the plugin, we see the topic name text box with **`/cmd_vel`**
- Here we must write the correct topic used to control the Thymio
  - **What do you think it will be?**

# Exercise 3: simulation time

Until now CoppeliaSim and the ROS nodes used two separate clocks.

This can be inconvenient if you plan to work only in CoppeliaSim and you for some reason *cough cough HW2* might need to keep a clock in sync with the physics simulation.

To do so we need to

1. Import a particular model into Coppelia
2. Set a parameter for all nodes from the launch file
3. Re-build our package

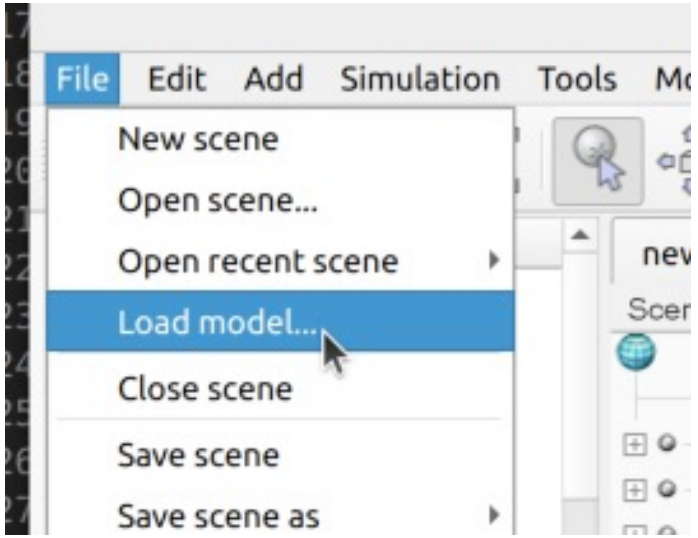
# Exercise 3: simulation time

Here you can find a .ttm model file

https://www.icorsi.ch/mod/resource/view.php?id=1040436

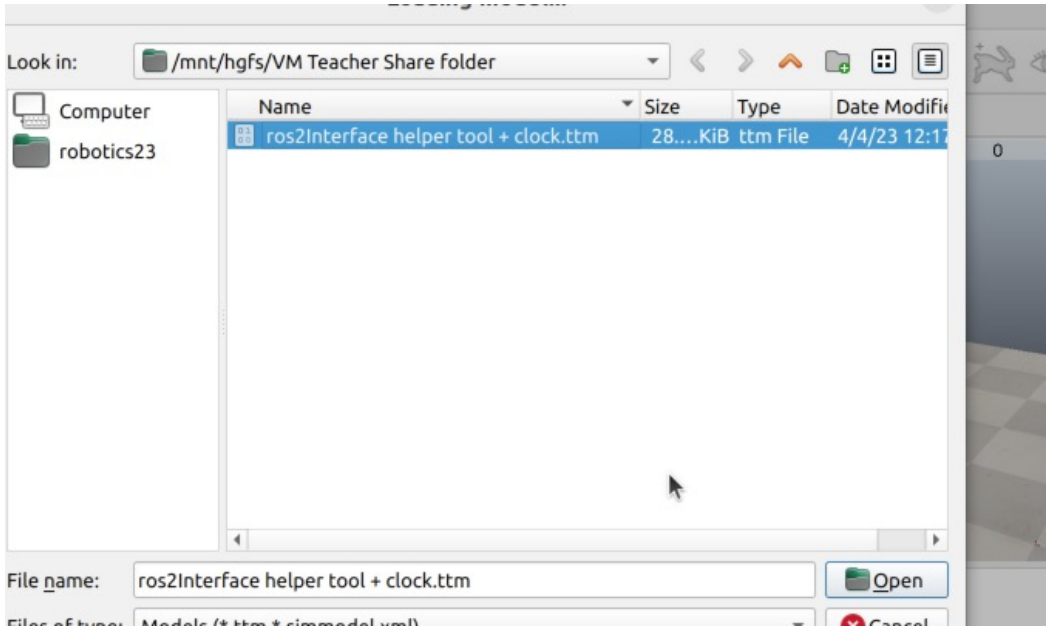
Download it and place it inside your Ubuntu machine (e.g. on the Desktop or in a shared folder, not in ~/dev_ws!).

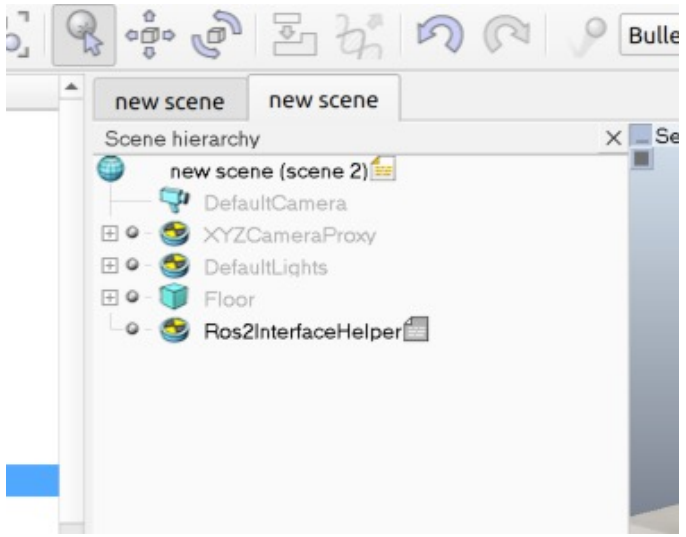
Load the model in Coppelia



# Exercise 3: simulation time

Navigate to the folder, then select the file and click “Open”. You should see it as in the right figure



# Exercise 3: simulation time

To set a parameter for all nodes in a package (guide here, select XML for the example)

1. Open the launch file (we suggest to use XML) using Visual Studio Code, for example the **`controller.launch.xml`** of the thymio_example package located under **`/home/robotics23/dev_ws/src/thymio_example/launch`**
2. Add the following lines in the launch file, BEFORE THE <node> TAG

```
<set_parameter name="use_sim_time" value="true" />

<!-- 'use_sim_time' will be set on all nodes following this line -->
```

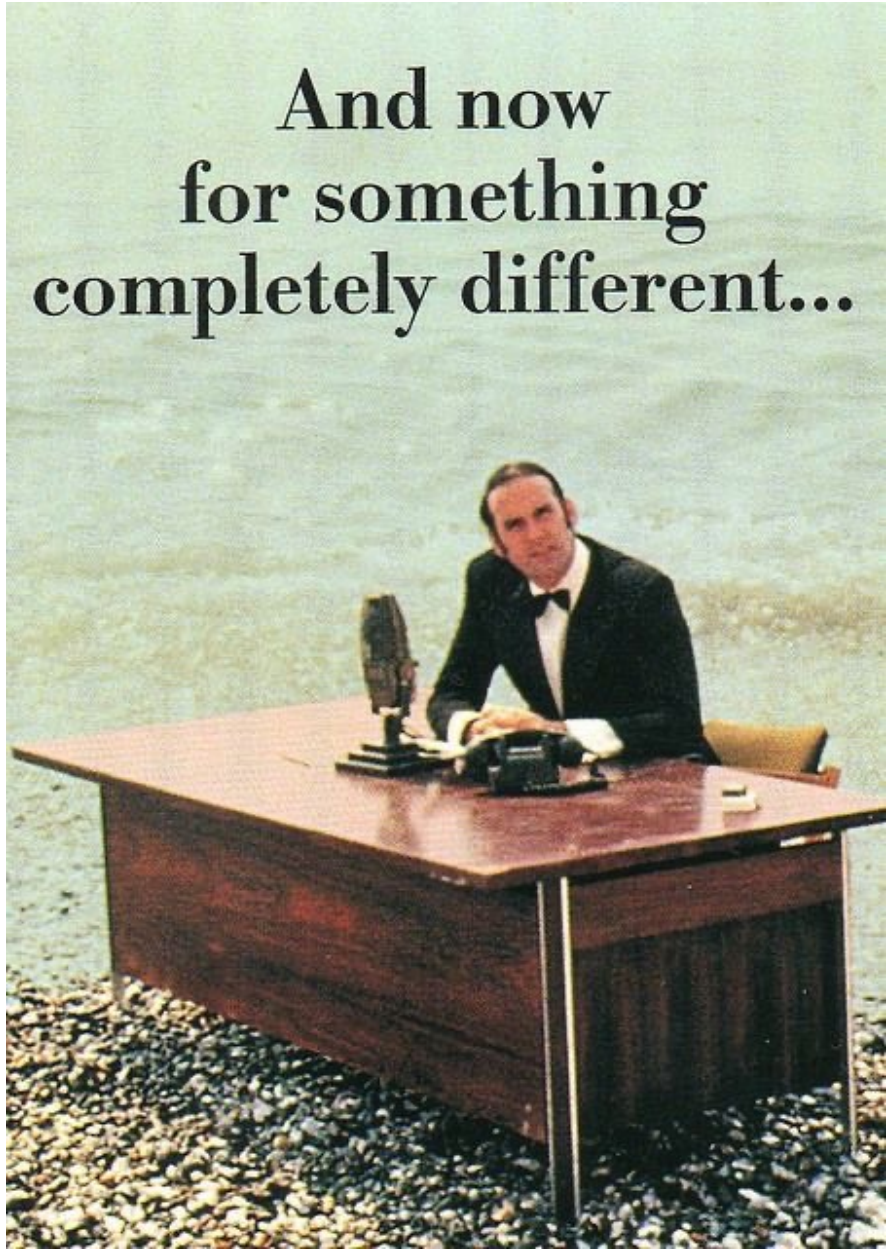
Note that the value of the *use_sim_time* param can be set to false when dealing with real robot and not a simulation but the package must be rebuilt to apply the change

3. Now in the code all the timers will follow the clock from the sim

# Exercise 3: WARNING

Not all packages use the sim time or are compatible with the param.

Avoid to speed up or slow down too much the sim time; in case you see sync problems use the real-time speed


And now
for something
completely different...

The breeze blows on the palm,

The sun shines on the blue,

Open Python, be calm,

This is the world of homework two

Introduction to ROS - Part 3

# Homework 2

- Today we are giving you the second Homework
- Link: https://www.icorsi.ch/mod/assign/view.php?id=1040470
- **Groups are allowed (max 2 students)** go at the link and choose yours https://www.icorsi.ch/mod/choicegroup/view.php?id=1040469
- The deadline is **April 23 at 23:59**
- This time you will have to control the simulated MightyThymio in CoppeliaSim
- NOTE: after submitting the assignment, you must answer the questionnaire https://www.icorsi.ch/mod/feedback/view.php?id=1040471
- NOTE 2: we will do a support class on April 19

USI/SUPSI

IDSIA

Università della Svizzera italiana

# Good luck with HW2

Elia Cereda, Simone Arreghini