



GAZEBO

**Manuel Ferre y  
Paloma de la Puente**



**Centre for Automation and Robotics**

*Centro de Automática y Robótica*

Madrid, Spain

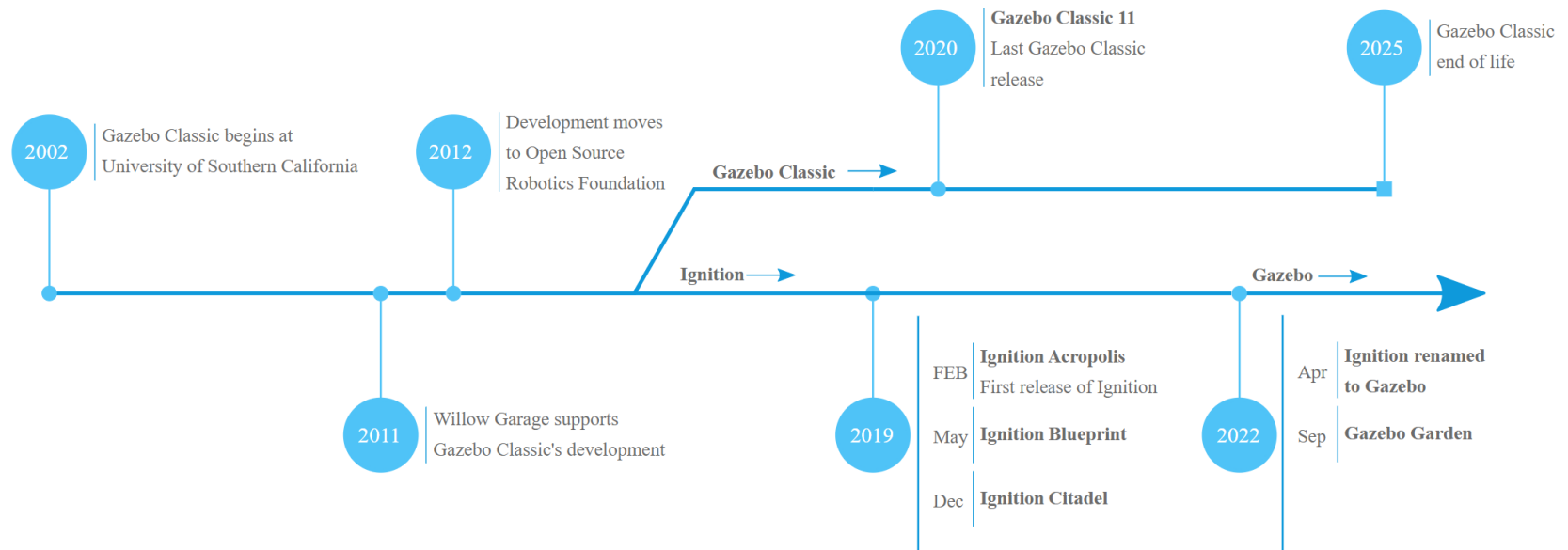
1. Introducción
2. Instalación de Gazebo
3. Simulador Gazebo stand-alone
4. Crear simulaciones con Gazebo
5. Conexión Gazebo – ROS2

# Introducción Gazebo

# Introducción

El siguiente gráfico muestra la evolución de Gazebo desde su primer lanzamiento en 2002, en estos momentos se puede instalar:

- Gazebo Classic 11, la versión original del simulador
- Ignition, la versión actualizada de Gazebo que arrancó en 2019, y
- Gazebo Garden, la continuación de Ignition que cambió de nombre.



## Combinaciones entre ROS y Gazebo

# ROS



Noetic



Foxy



Galactic



Humble



Rolling

# GAZEBO



Citadel



Edifice



Fortress



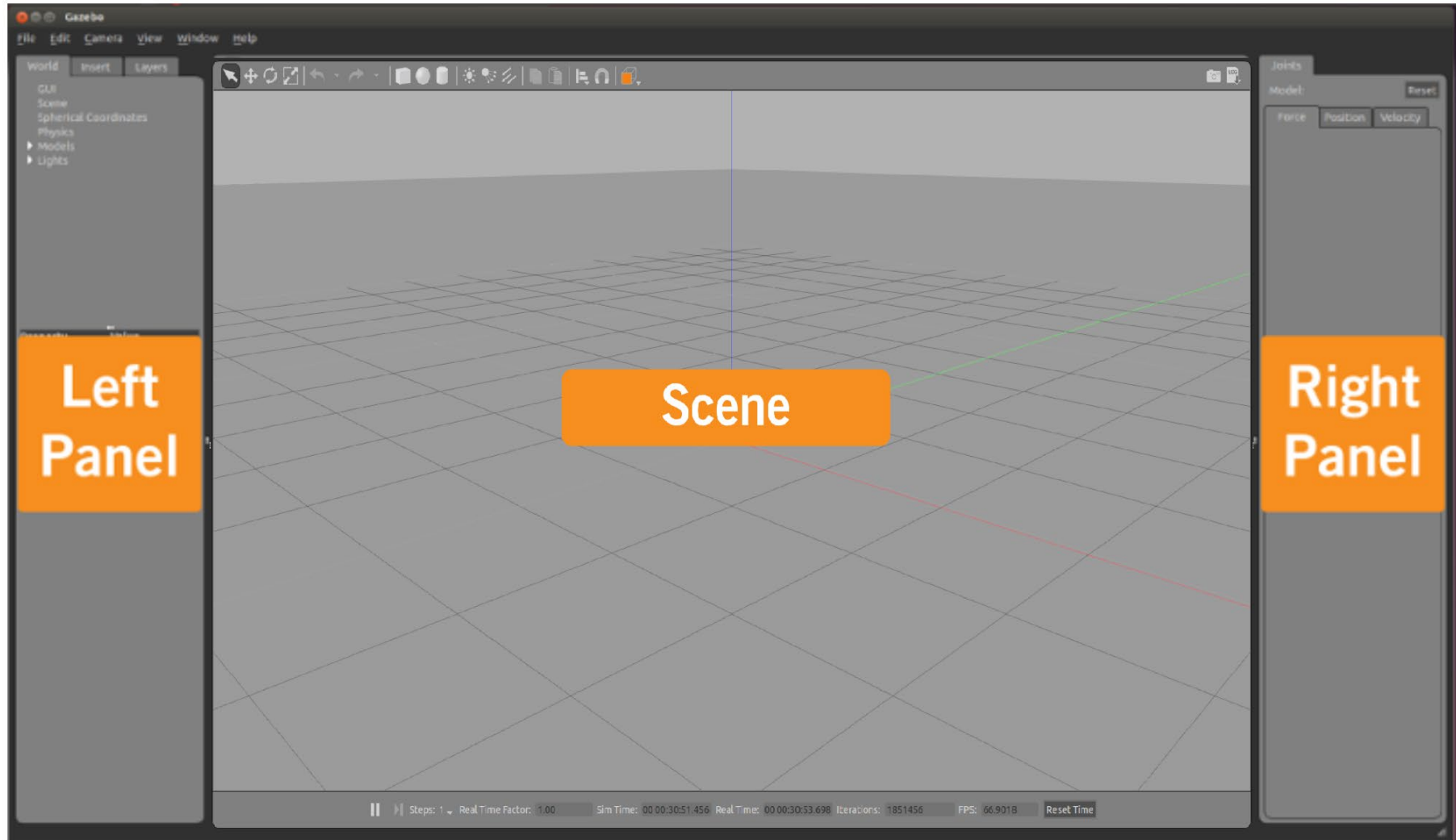
Garden

— binaries available  
- - - only from source

\*Binary packages indicate recommended combinations

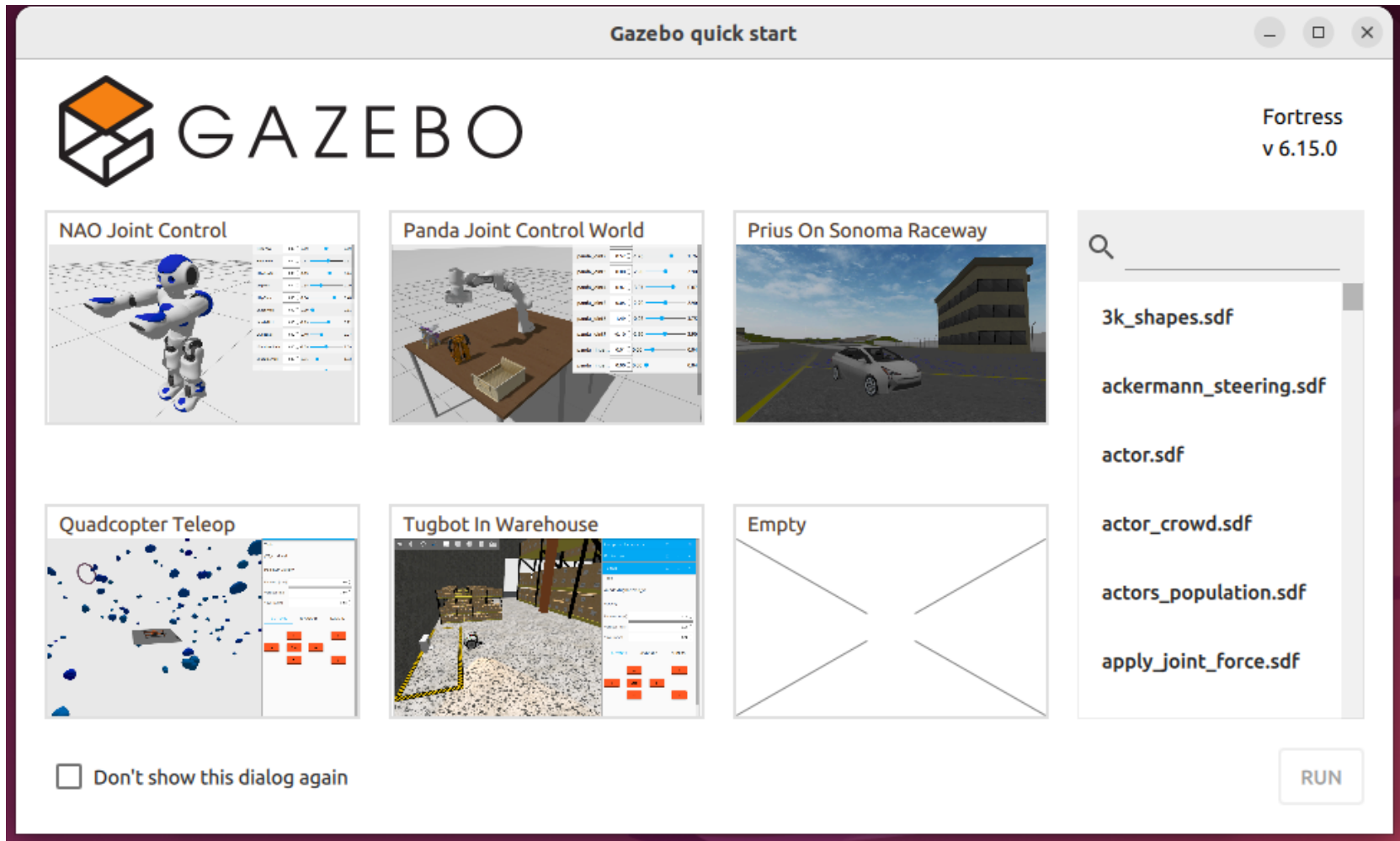
# Introducción

Aspecto de gazebo 11 utilizado en ROS

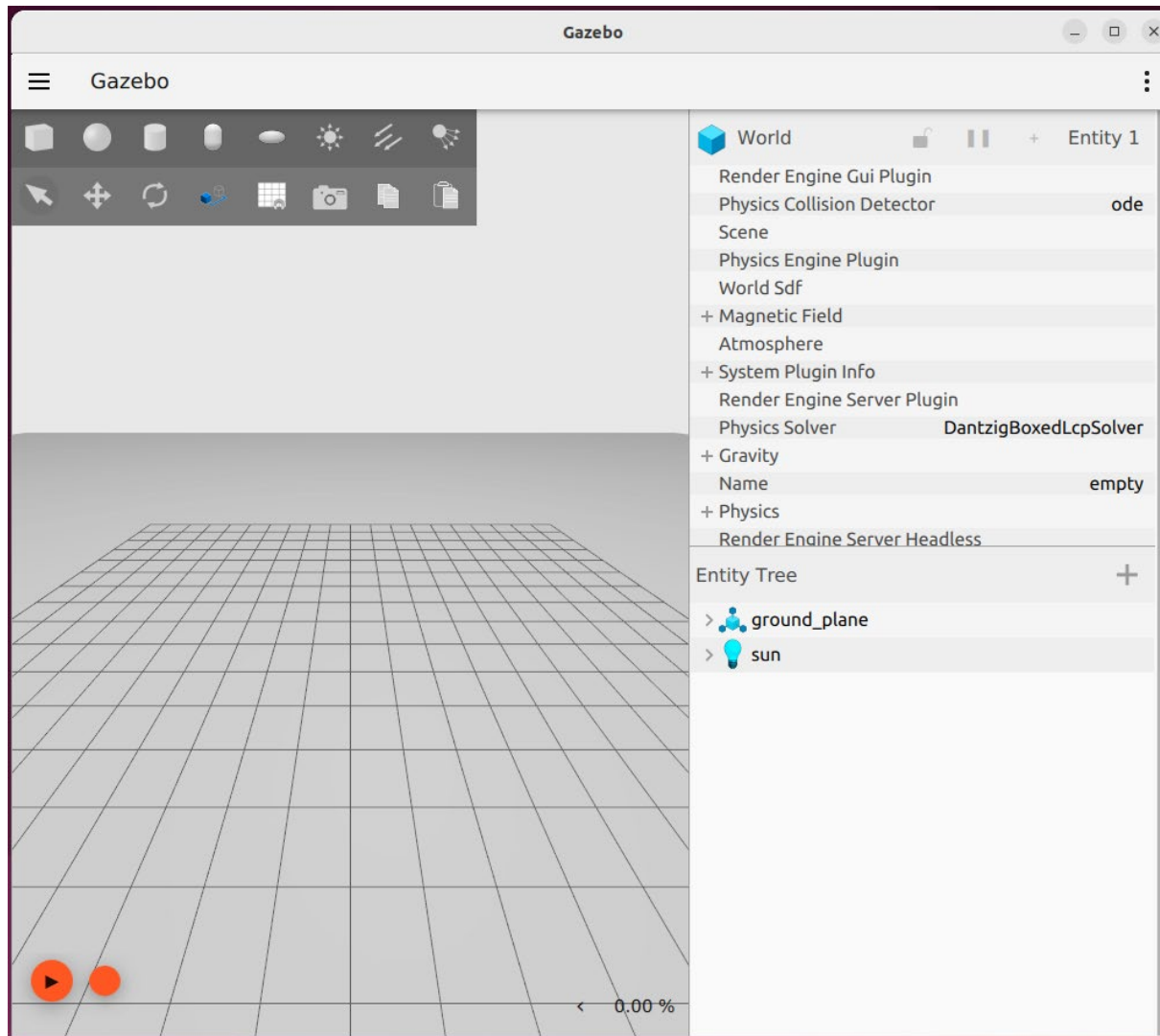


# Introducción

## Aspecto de ignition



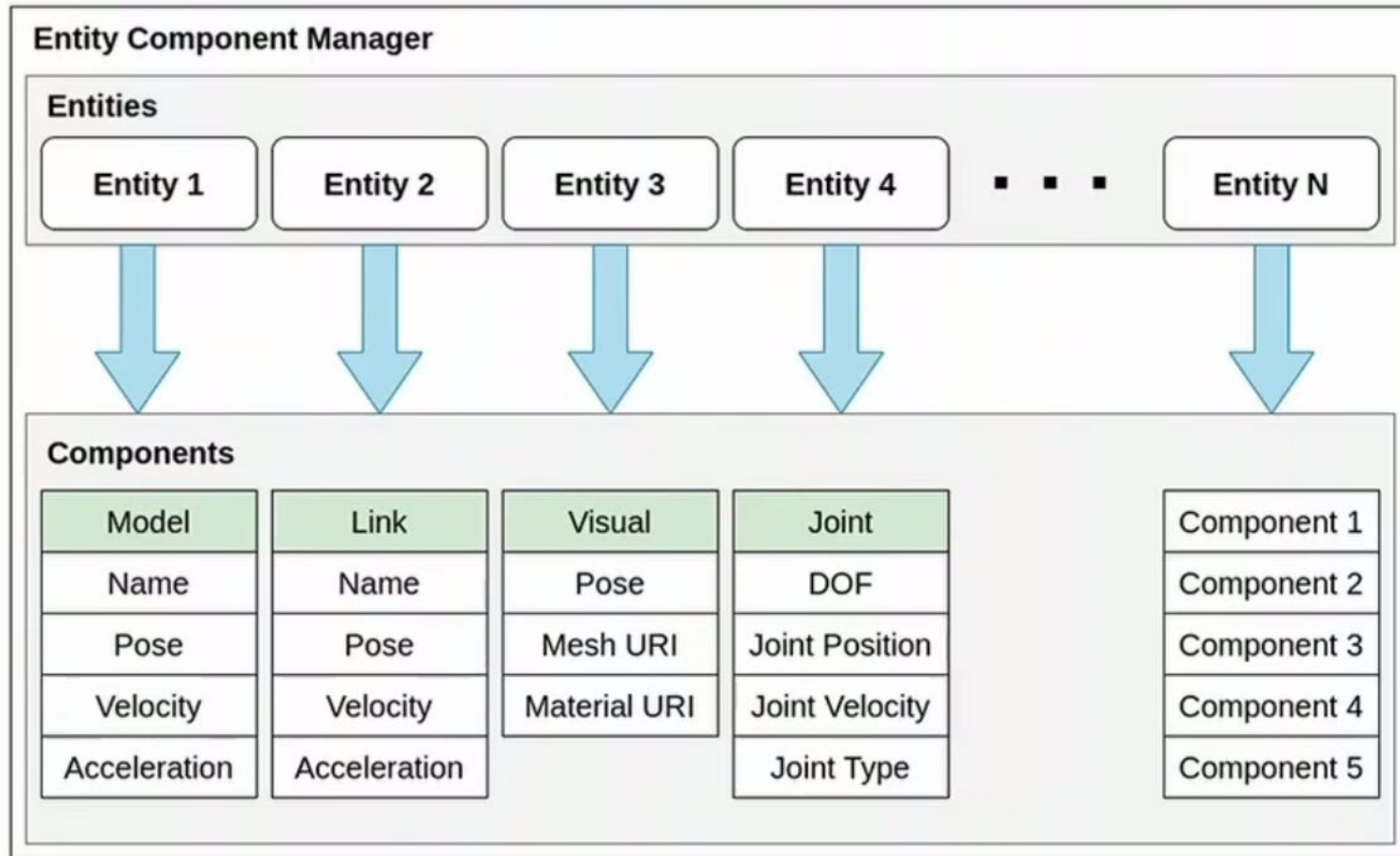
## Aspecto de ignition





# Introducción

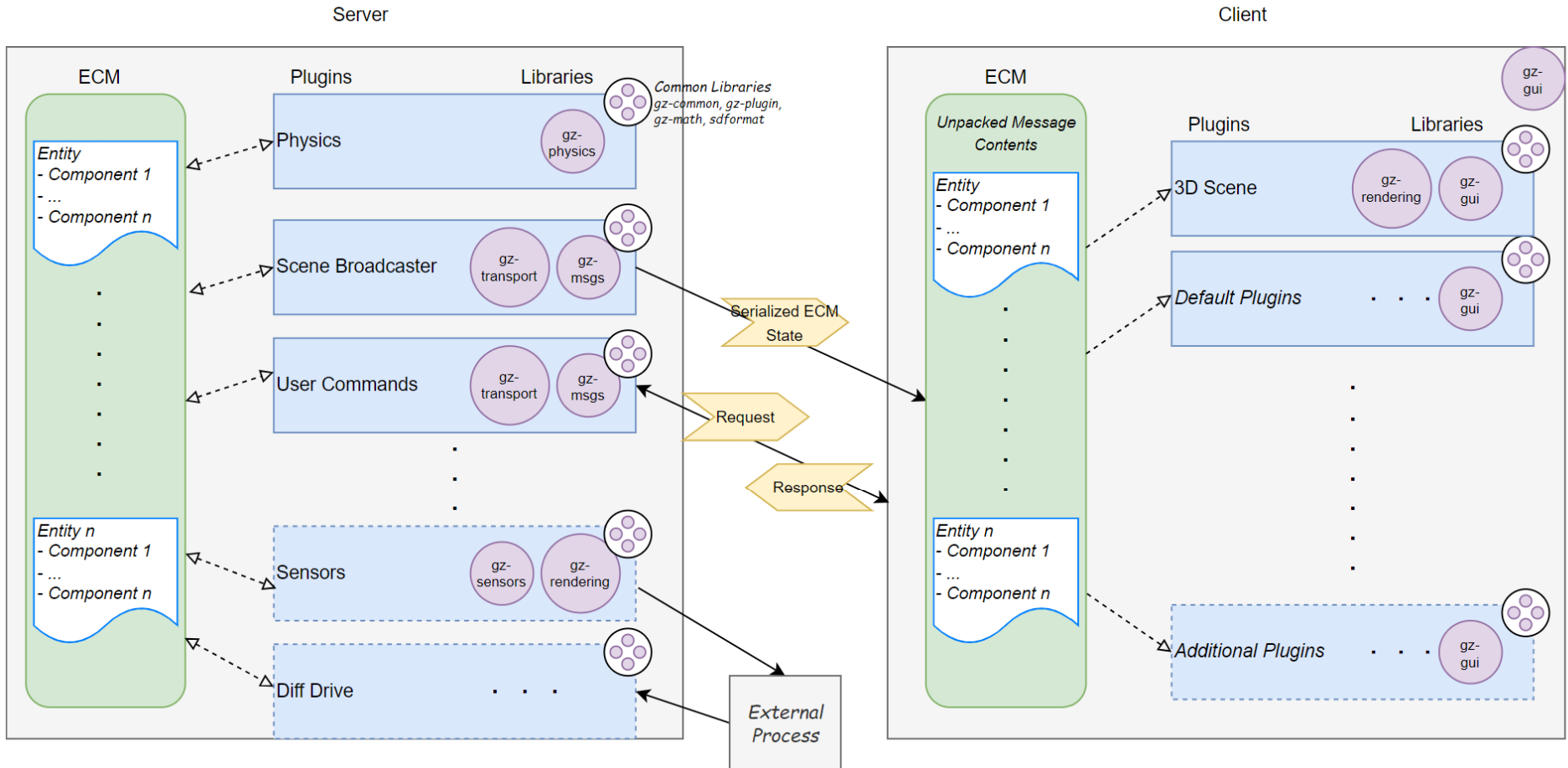
Arquitectura cliente/servidor de Ignition Gazebo, basada en entidades.



# Introducción

## Arquitectura cliente/servidor de Ignition Gazebo.

Gazebo Sim Architecture



1. Introducción
2. Instalación de Gazebo
3. Simulador Gazebo stand-alone
4. Crear simulaciones con Gazebo
5. Conexión Gazebo – ROS2

# Instalación Gazebo

# Instalación de Gazebo

El siguiente cuadro muestra el mejor rendimiento para cada versión de Ubuntu y Gazebo

	GZ Citadel (LTS)	GZ Fortress (LTS)	GZ Garden	GZ Harmonic (LTS)
ROS 2 Jazzy (LTS) <sup>1</sup>	✗	✗	⚡	✓
ROS 2 Rolling	✗	✓	⚡	⚡
ROS 2 Iron	✗	✓	⚡	⚡
ROS 2 Humble (LTS)	✗	✓	⚡	⚡
ROS 2 Foxy (LTS)	✓	✗	✗	✗
ROS 1 Noetic (LTS)	✓	⚡	✗	✗


- ✓ - Recommended combination
- ✗ - Incompatible / not possible.
- ⚡ - Possible, *but use with caution*. These combinations of ROS and Gazebo can be made to work together, but some effort is required.

[https://gazebo-sim.org/docs/latest/ros\\_installation](https://gazebo-sim.org/docs/latest/ros_installation)

# Instalación de Gazebo

<https://gazebo.sim.org/docs/fortress/install> (todas las librerías)

En caso de instalar con ROS se recomienda otra instalación



Gazebo Sim  
API Reference  
8.0.0

- Tutorials
- Classes
- Namespaces
- Files
- Gazebo Website

## Tutorials

Welcome to the Gazebo Sim tutorials. These tutorials will guide you through the process of understanding the capabilities of the Gazebo Sim library and how to use the library effectively.

## Getting Started

- [Installation](#): Install instructions.
- [Terminology](#): List of terms used across the documentation.
- [GUI configuration](#): Customizing your layout.
- [Server configuration](#): Customizing what system plugins are loaded.
- [Model Command](#): Use the CLI to get information about the models in a simulation.
- [Reset simulation](#): Reset simulation
- [Finding resources](#): The different ways in which Gazebo looks for files.
- [Debugging](#): Information about debugging Gazebo.

## GUI and rendering features

- [Move Camera to model](#): Move camera to model
- [Model Photo Shoot](#): Taking perspective, top, front, and side pictures of a model.
- [Video Recorder](#): Record videos from the 3D render window.
- [Headless rendering](#): Access the GPU on a remote machine to produce sensor data without an X server.
- [Apply Force and Torque](#): Applying forces and/or torques to models during simulation through the GUI.
- [Mouse Drag](#): Move models by dragging them in the scene using forces and torques.

## INSTALACIÓN DE GAZEBO CON ROS

### Instalación en función de la versión de ROS

*\$ sudo apt-get install ros- $\{ROS\_DISTRO\}$ -ros-gz*

*$\{ROS\_DISTRO\}$  == humble, rolling, foxy, noetic, etc.*

### Configuración:

- Ubuntu Jammy 22.04
- ROS 2 Humble Hawksbill
- Gazebo Fortress (Gazebo Sim, version 6.15.0)

[https://gazebo-sim.org/docs/latest/ros\\_installation](https://gazebo-sim.org/docs/latest/ros_installation)

1. Introducción
2. Instalación de Gazebo
3. Simulador Gazebo stand-alone
4. Crear simulaciones con Gazebo
5. Conexión Gazebo – ROS2



## Arrancar Gazebo

```
ign gazebo
```

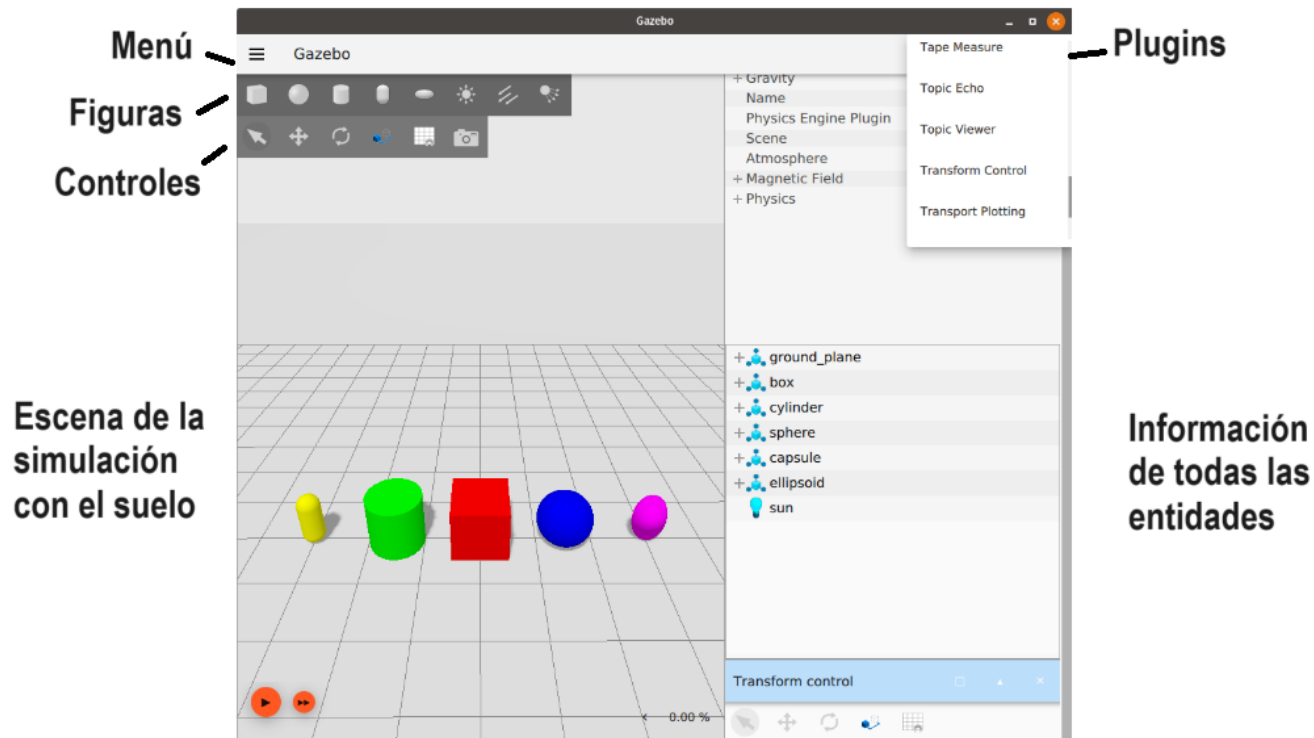
Incluir varios objetos en la simulación.

Posteriormente guardar el mundo (ejemplo1.sdf), para utilizarlo posteriormente.

```
ign gazebo ejemplo1.sdf
```

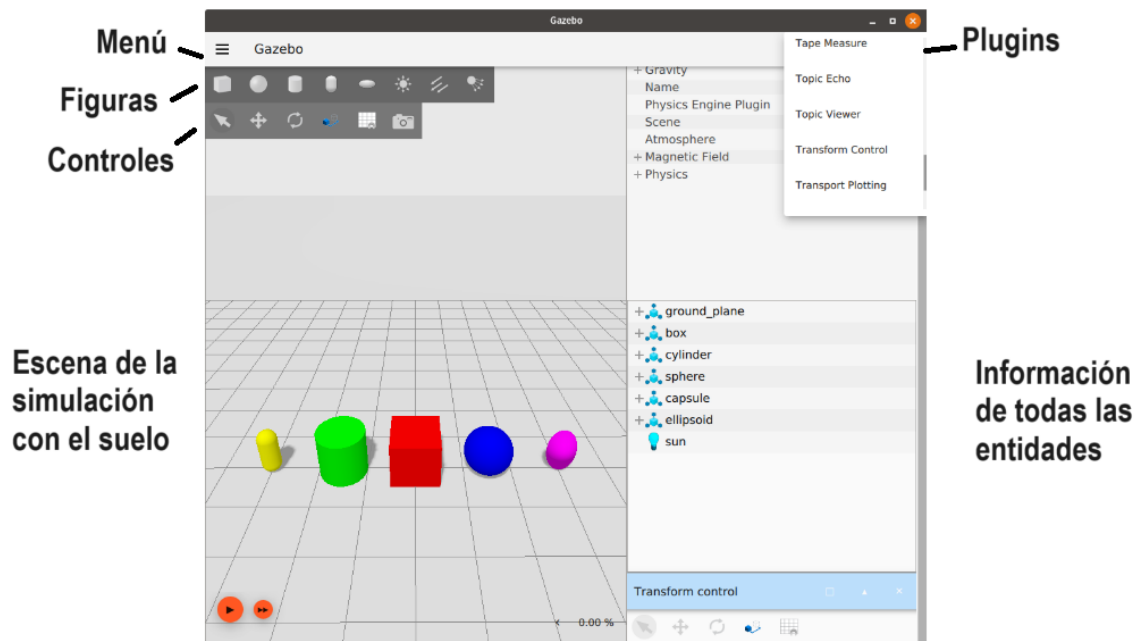
## Arrancar Gazebo

```
ign gazebo shapes.sdf
```



## Ejercicios:

- Manipular los objetos con los controles
- Cambiar la representación del suelo con el plugin 'Grid config'



# Tutoriales

[FEATURES](#)[SHOWCASE](#)[DOCS](#)[COMMUNITY](#)[MORE](#)[APP](#)

## Docs / Gazebo Fortress

Supported Sep, 2021 to Sep, 2026

LTS

Release

Fortress (LTS)

### Get Started

> [Install](#)

> [Tutorials](#)

[Feature Comparison](#)

[Ros/gazebo Installation](#)

[Roadmap](#)

[Release Features](#)

[Releases](#)

> [Development](#)

[Sim Architecture](#)

> [Library Reference](#)

## Ignition Tutorials

These tutorials cover general concepts to help get you started with Ignition.

### Basics tutorials

- [Building Your Own Robot](#)
- [Moving the Robot](#)
- [SDF Worlds](#)
- [Sensors](#)
- [Actors](#)

### GUI tutorials

- [Understanding the GUI](#)
- [Manipulating Models](#)
- [Model Insertion from Fuel](#)
- [Keyboard Shortcuts](#)

Ignition Tutorials

Basics tutorials

GUI tutorials

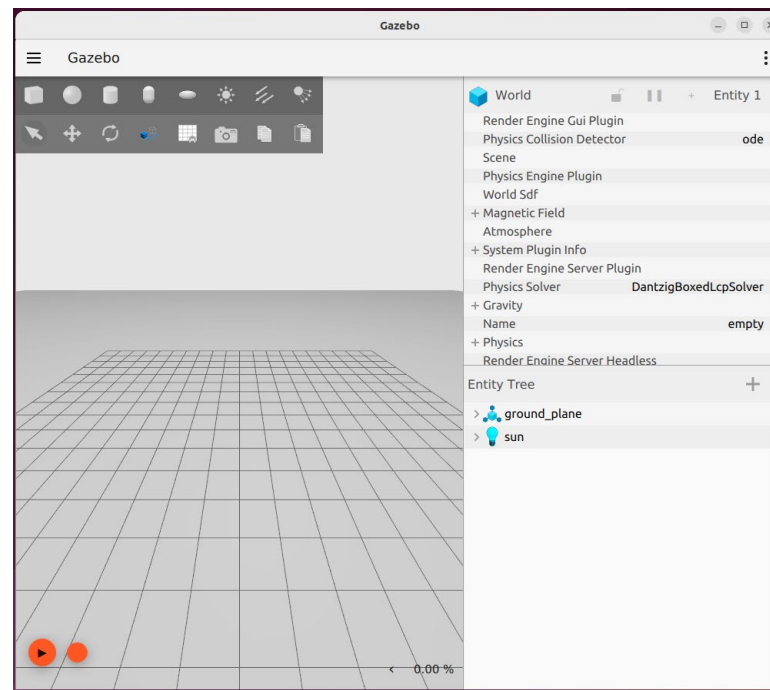
ROS integration

Per-library tutorials

<https://gazebo-sim.org/docs/fortress/tutorials>

Copiar el fichero: `building_robot.sdf`

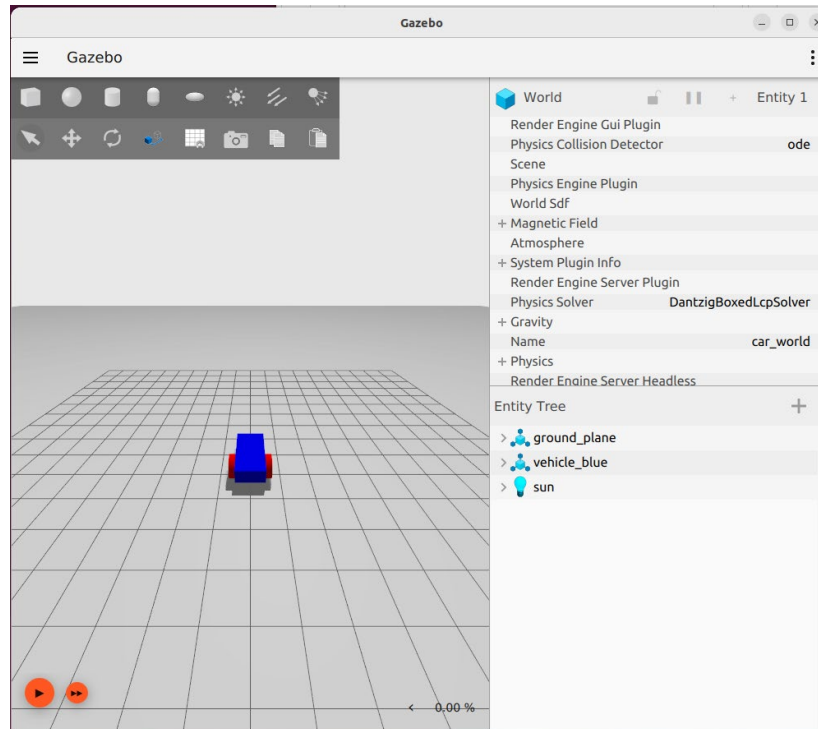
Ejecutar: `ign gazebo building_robot.sdf`



[https://gazebo-sim.org/docs/fortress/building\\_robot](https://gazebo-sim.org/docs/fortress/building_robot)

Copiar el fichero: ejemplo.sdf

Ejecutar: `ign gazebo ejemploCoche.sdf`



[https://gazebo-sim.org/docs/fortress/building\\_robot](https://gazebo-sim.org/docs/fortress/building_robot)

# SDF: Simulation Description Format

## Components of SDF Models

**Links:** A link contains the physical properties of one body of the model. This can be a wheel, or a link in a joint chain. Each link may contain many collision and visual elements. Try to reduce the number of links in your models in order to improve performance and stability. For example, a table model could consist of 5 links (4 for the legs and 1 for the top) connected via joints. However, this is overly complex, especially since the joints will never move. Instead, create the table with 1 link and 5 collision elements.

**Collision:** A collision element encapsulates a geometry that is used for collision checking. This can be a simple shape (which is preferred), or a triangle mesh (which consumes greater resources). A link may contain many collision elements.

**Visual:** A visual element is used to visualize parts of a link. A link may contain 0 or more visual elements.

**Inertial:** The inertial element describes the dynamic properties of the link, such as mass and rotational inertia matrix.

**Sensor:** A sensor collects data from the world for use in plugins. A link may contain 0 or more sensors.

**Light:** A light element describes a light source attached to a link. A link may contain 0 or more lights.

**Joints:** A joint connects two links. A parent and child relationship is established along with other parameters such as axis of rotation, and joint limits.

**Plugins:** A plugin is a shared library created by a third party to control a model.



## COMPONENTES DE UN MODELO SDF

### Links

Propiedades físicas de un objeto del modelo

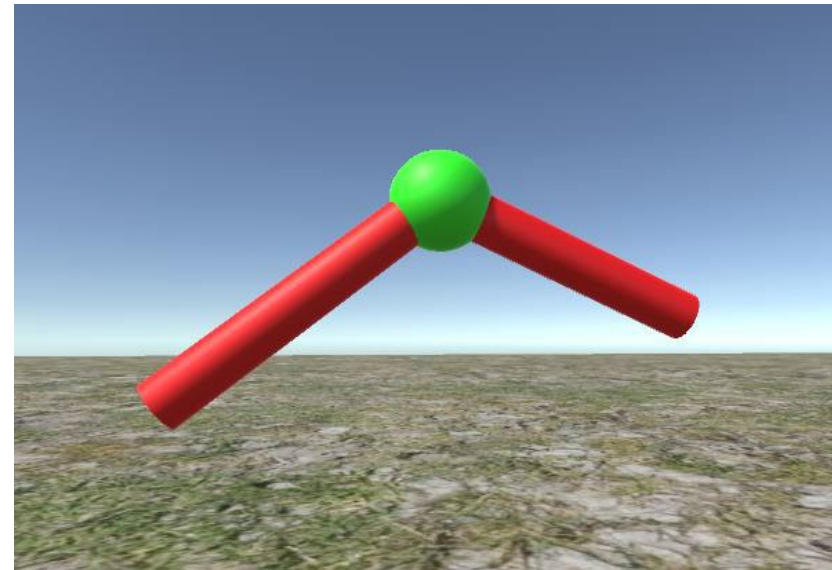
(Collision, Visual, Inertial, Sensor).

### Joints

Conector entre dos o más Links.

### Plugins

Código que realiza el control del modelo.



# Formatos de ficheros

Existe un gran número de simuladores para robots y cada uno de ellos tiene su propio formato nativo de datos, y admite otros formatos de lectura de ficheros 3D.

Simulador	Programación	Formatos que utiliza	Entorno
<b>Gazebo</b>	C++	SDF, URDF, OBJ, STL, Collada	Ubuntu / ROS
<b>RoboDK</b>	Python, ...	SLDPRT, SLDASM, STEP, OBJ, STL, 3DS, Collada, VRML, URDF, ...	Windows, y otras plataformas
<b>Webots</b>	C++	WBT, VRML, X3D, 3DS, Blender, BVH, Collada, FBX, STL, OBJ, URDF, ...	Windows
<b>Coppelia/Vrep</b>	C, Java, LUA	IRDF, Collada, DXF, ...	Ubuntu, Windows

Copiar el fichero: ejemplo.sdf

Ejecutar: `ign gazebo ejemploCoche.sdf`

EJERCICIO: añadir 1 ruedas esférica adicional en la parte frontal del vehículo

[https://gazebosim.org/docs/fortress/building\\_robot](https://gazebosim.org/docs/fortress/building_robot)

1. Introducción
2. Instalación de Gazebo
3. Simulador Gazebo stand-alone
4. Crear simulaciones con Gazebo
5. Conexión Gazebo – ROS2

El formato SDF se utiliza para la creación de mundos en Gazebo.

```
<?xml version="1.0" ?>
<sdf version="1.8">
  <world name="world_demo">
    ...
    ...
  </world>
</sdf>
```

[https://gazebo-sim.org/docs/fortress/sdf\\_worlds](https://gazebo-sim.org/docs/fortress/sdf_worlds)

Los componentes más importantes son:

- Motor físico: determina la interacción
- Plugins: código de ejecución
- Light: iluminación de la simulación
- Añadir modelos: desde Fuel

```
<include>  
  <uri>  
    https://fuel.gazebosim.org/1.0/OpenRobotics/models/Coke  
  </uri>  
</include>
```

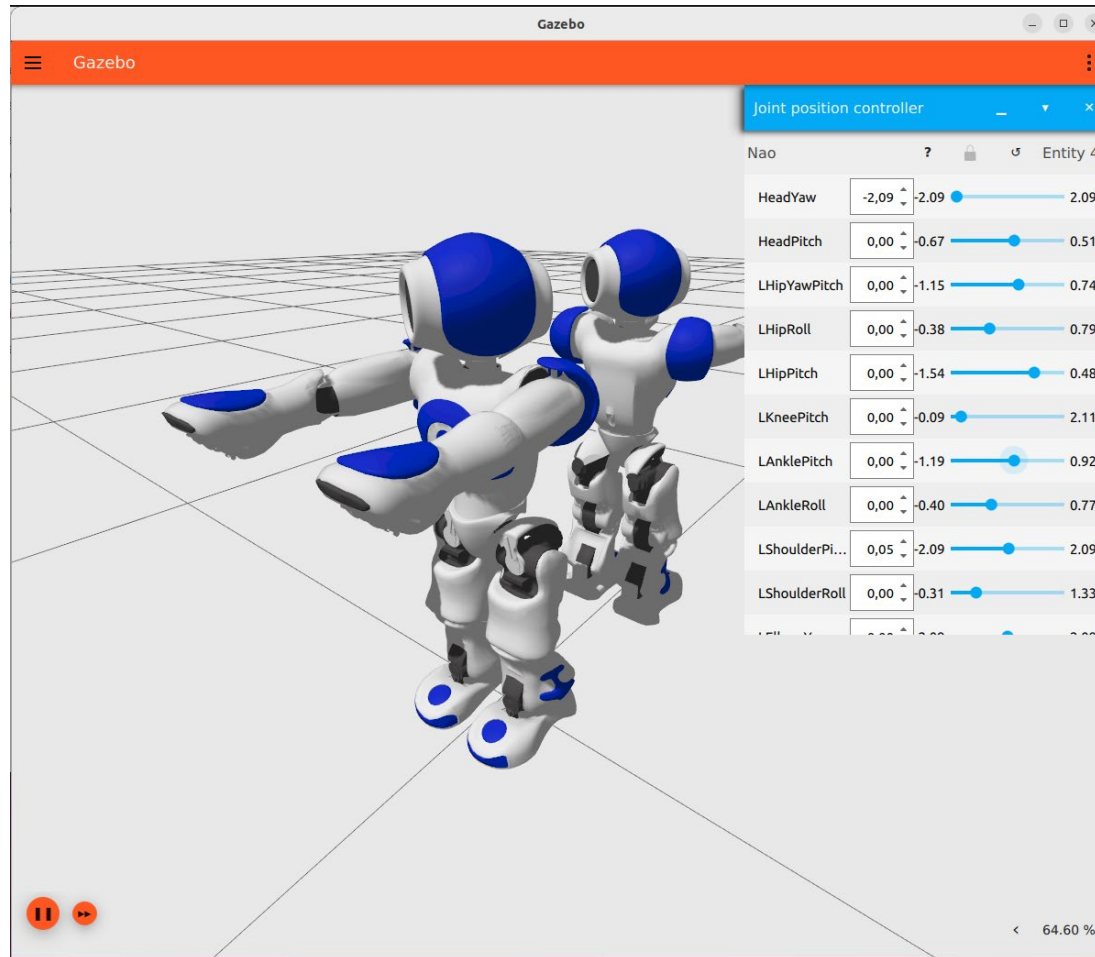
[https://gazebosim.org/docs/fortress/sdf\\_worlds](https://gazebosim.org/docs/fortress/sdf_worlds)

Los componentes más importantes son:

- GUI: La interfaz es configurable, y ofrece:
  - Estadísticas
  - Árbol de entidades
  - Tamaño de la ventana
  - Etc....

[https://gazebo-sim.org/docs/fortress/sdf\\_worlds](https://gazebo-sim.org/docs/fortress/sdf_worlds)

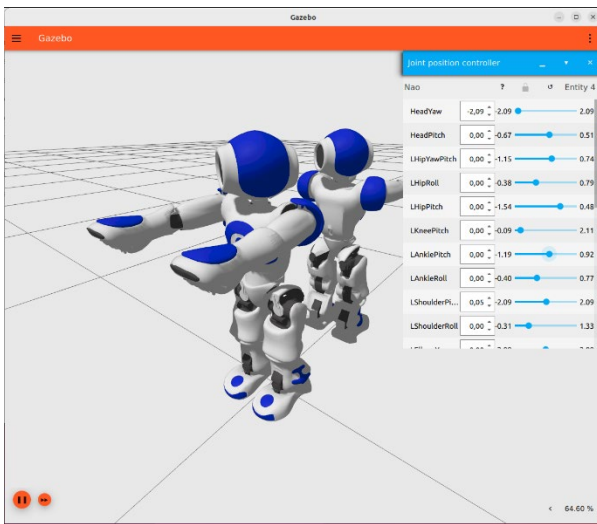
EJERCICIO: arrancar gazebo con el ejemplo del Nao y crear un nuevo mundo con 2 robots Nao





EJERCICIO: arrancar gazebo con el ejemplo del Nao y crear un nuevo mundo con 2 robots Nao

## SOLUCIÓN



```
<plugin name='JointPositionController2' filename='JointPositionController'>
  <ignition-gui>
    <property type='double' key='height'>600</property>
    <property type='double' key='width'>400</property>
    <property type='string' key='state'>floating</property>
    <anchors target='3D View'>
      <line own='right' target='right'/>
      <line own='top' target='top'/>
    </anchors>
  </ignition-gui>
  <model_name>Nao2</model_name>
</plugin>
```

```
<include>
  <uri>https://fuel.ignitionrobotics.org/1.0/OpenRobotics/models/NAO with Ignition position controller</uri>
  <name>Nao2</name>
  <pose>-0.3 0 0.35 0 -0 3.14</pose>
</include>
```

1. Introducción
2. Instalación de Gazebo
3. Simulador Gazebo stand-alone
4. Crear simulaciones con Gazebo
5. Conexión Gazebo – ROS2

El siguiente código implementa el código para mover el robot con 2 ruedas direccionales

```
<plugin
  filename="libignition-gazebo-diff-drive-system.so"
  name="ignition::gazebo::systems::DiffDrive">
  <left_joint>left_wheel_joint</left_joint>
  <right_joint>right_wheel_joint</right_joint>
  <wheel_separation>1.2</wheel_separation>
  <wheel_radius>0.4</wheel_radius>
  <odom_publish_frequency>1</odom_publish_frequency>
  <topic>cmd_vel</topic>
</plugin>
```

[https://gazebosim.org/docs/fortress/moving\\_robot](https://gazebosim.org/docs/fortress/moving_robot)

## Ejecutar la simulación del robot en un terminal

```
ign gazebo mover_robot.sdf
```

## Desde otro terminal se controla la velocidad del vehículo

```
ign topic -t "/cmd_vel" -m ignition.msgs.Twist -p  
"linear: {x: 0.5}, angular: {z: 0.05}"
```

[https://gazebosim.org/docs/fortress/moving\\_robot](https://gazebosim.org/docs/fortress/moving_robot)

Ejercicio: añadir objetos y que choque el vehículo

[https://gazebo-sim.org/docs/fortress/moving\\_robot](https://gazebo-sim.org/docs/fortress/moving_robot)

Ejercicio: mover el robot desde el teclado

Revisar el tutorial y seguir las instrucciones

En el siguiente tutorial describe cómo añadir sensores al vehículo y al entorno, en concreto:

- IMU
- Contact sensor
- Lidar

<https://gazebosim.org/docs/fortress/sensors>

## Sensor IMU:

- 1. Añadir el plugin de la IMU

```
<plugin filename="libignition-gazebo-imu-system.so"  
        name="ignition::gazebo::systems::Imu">  
</plugin>
```

- 2. Añadir el sensor al link del chasis (al final)

```
<sensor name="imu_sensor" type="imu">  
  <always_on>1</always_on>  
  <update_rate>1</update_rate>  
  <visualize>true</visualize>  
  <topic>imu</topic>  
</sensor>
```



## Sensor IMU:

- 3. Ejecutar la simulación y pulsar 'Play'

```
ign gazebo sensor_imu.sdf
```

- 4. Visualizar la información de la IMU

```
ign topic -e -t /imu
```

## Sensor de contacto en un muro:

### - 1. Añadir el objeto del muro al mundo

```
<model name='wall'>
  <static>true</static>
  <pose>5 0 0 0 0 0</pose><!--pose relative to the world-->
  <link name='box'>
    <pose/>
    <visual name='visual'>
      <geometry>
        <box>
          <size>0.5 10.0 2.0</size>
        </box>
      </geometry>
      <!--let's add color to our link-->
      <material>
        <ambient>0.0 0.0 1.0 1</ambient>
        <diffuse>0.0 0.0 1.0 1</diffuse>
        <specular>0.0 0.0 1.0 1</specular>
      </material>
    </visual>
    <collision name='collision'>
      <geometry>
        <box>
          <size>0.5 10.0 2.0</size>
        </box>
      </geometry>
    </collision>
  </link>
</model>
```

Sensor de contacto en un muro:

- 2. Añadir el plugging y el sensor de contacto al muro

```
<plugin filename="libignition-gazebo-contact-system.so"  
      name="ignition::gazebo::systems::Contact">  
</plugin>
```

```
<sensor name='sensor_contact' type='contact'>  
  <contact>  
    <collision>collision</collision>  
  </contact>  
</sensor>
```

## Sensor de contacto en un muro:

### - 3. Añadir el plugging de detección de contacto

```
<plugin filename="libignition-gazebo-touchplugin-system.so"  
  name="ignition::gazebo::systems::TouchPlugin">  
  <target>vehicle_blue</target>  
  <namespace>wall</namespace>  
  <time>0.001</time>  
  <enabled>true</enabled>  
</plugin>
```

## Sensor de contacto en un muro:

### - 4. Añadir el plugging de respuesta al evento

```
<plugin filename="libignition-gazebo-triggered-publisher-system.so"
  name="ignition::gazebo::systems::TriggeredPublisher">
  <input type="ignition.msgs.Boolean" topic="/wall/touched">
    <match>data: true</match>
  </input>
  <output type="ignition.msgs.Twist" topic="/cmd_vel">
    linear: {x: 0.0}, angular: {z: 0.0}
  </output>
</plugin>
```

Sensor de contacto en un muro:

- 5. Ejecutar los programas

```
ign gazebo sensor_contacto.sdf
```

```
ign topic -e -t /wall/touched
```