

Human-Robot Interaction and Autonomous Systems

AI 616

Master of Science in Artificial intelligence

Prepared by Dr. Khaled Eskaf

Lecture 3





مجموعة جامعة
الجميع الذكية
للتعليم القابضة

MIDOCEAN
UNIVERSITY

Human-Robot Interaction and Autonomous Systems

Lecture 3

AI 616

Prepared by
Dr. Khaled Eskaf

www.midocean.ae



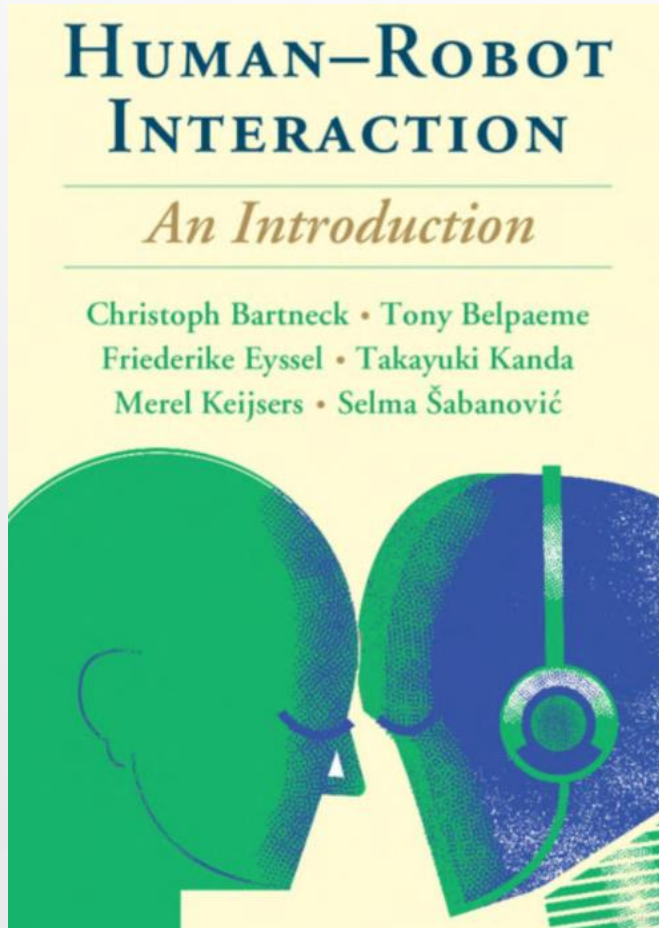
Course Outlines

Module	Content	Simulation Lab
1. Introduction to Robotics	- Fundamentals of Robot Control - Classification of Robotics - Exploration of Robot Types	- Basic Robot Control - Modeling Robot Classes - Simulating Various Robot Types
2. Advanced HRI Studies	- Introductory to Advanced Concepts in HRI	- Basic to Complex HRI Scenarios
3. Social Dynamics in Robotics	- Architectures for Social Interaction - Case Studies and History of HRI	- Social Interaction Simulations - Historical Evolution in HRI
4. Strategic HRI System Design	- Core Principles and Communication Strategies in HRI	- Designing Interfaces - Communication Protocol Simulations
5. HRI Efficiency and Effectiveness	- Metrics for Mission Effectiveness and Human Factors	- Implementing and Analyzing HRI Metrics
6. Innovative HRI System Design	- Exploring Innovative Designs and Real-World Applications	- Testing Innovative Designs - Simulating Diverse Scenarios
7. HRI System Classification	- Developing Comprehensive Classifications	- Classification Application and Case Studies
8. Human Factors in HRI	- Analyzing Ergonomics and Human Factors	- Ergonomic Simulations - Scenario-Based Human Factor Analysis
9. Robotics and AI Integration	- Theoretical Concepts and Practical AI Applications	- AI Algorithm Implementation - AI-Driven HRI Modeling
10. HRI Case Studies	- Mobile Assistive Robot Studies	- Designing and Testing Mobile Assistive Robots

Human-Robot Interaction An

Introduction

By [Christoph Bartneck](#), [Tony Belpaeme](#), [Friederike Eyszel](#), [Takayuki Kanda](#), [Merel Keijsers](#), [Selma Šabanović](#) · 2020



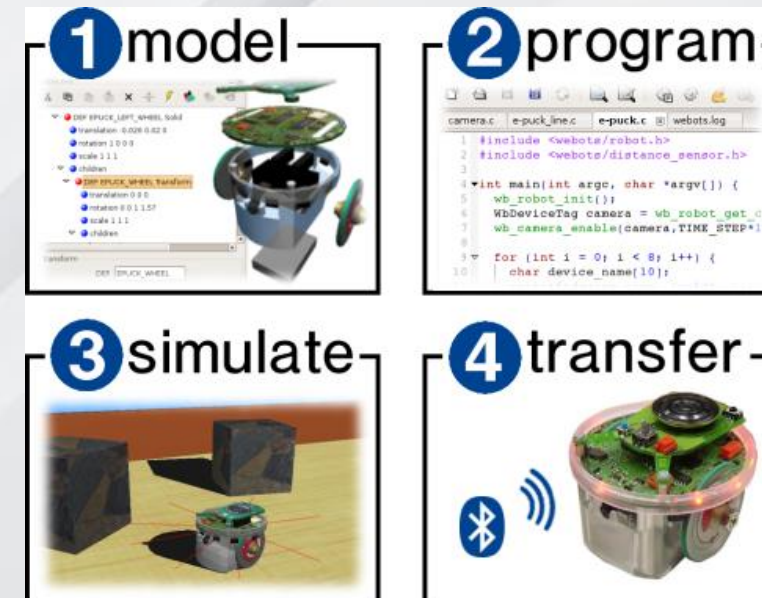
Webots

robot simulation

MIDOCEAN
UNIVERSITY

Webots is an open source and multi-platform desktop application used to **simulate** robots

<https://cyberbotics.com/>



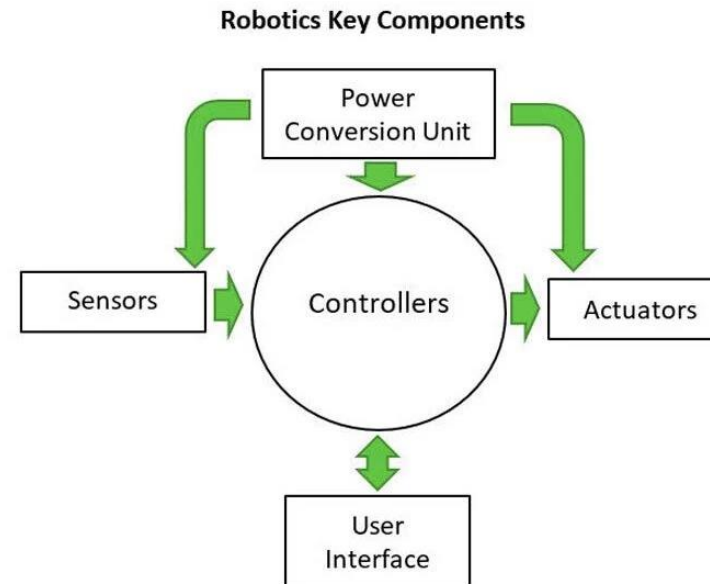
Chapter

02



The Core Components of a Robot

An illustrative diagram showing the robot's core components: Computer Parts, Mechanical Components, Sensors, and Embedded Systems, leading to Human-Robot Interaction.



The Core of Robotics : Understanding Computer Parts

Chapter

02

Computer Parts in Robotics

Simplified explanations of CPUs, Memory, and Storage, using metaphors like the robot's "brain" and "memory bank."



Chapter

02

Actuators: The Muscles of Robots

1. What Are Actuators?

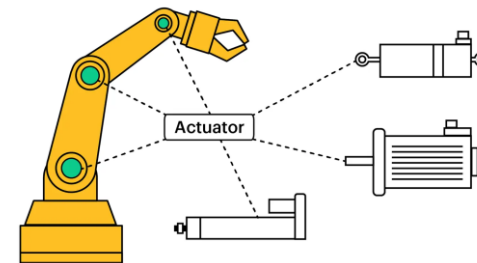
1. Brief Definition: Devices that convert electrical energy into physical motion.
2. Analogy: Just as muscles allow humans and animals to move, actuators enable robots to walk, grasp, lift, and perform other actions.

2. Types of Actuators

Electric Motors: The most common type, used in robotic arms, drones, and more. Illustrate with a snippet showing an electric motor in a robotic arm.

3. Why Actuators Are Important

Emphasize the role of actuators in expanding what robots can do, making them capable of interacting with the physical world in diverse and complex ways.



Chapter

02

Motors: Powering Robotic Movement

1. Introduction to Motors in Robotics

1. Brief explanation of how motors are the driving force behind robotic movement, acting as the "muscle" that powers everything from robotic arms to autonomous vehicles.

2. Types of Motors

1. Servo Motors:

1. Characterized by their ability to control angular position with precision. Commonly used in robotic arms, antennas, or any application where precise positioning is required.

2. Stepper Motors:

1. Known for their accuracy in controlling steps, making them ideal for applications requiring exact movements, such as 3D printers and CNC machines.

3. DC Motors:

1. Direct Current motors offer simple control and are used in a wide range of applications due to their speed variability and torque. Examples include hobby robots, drones, and electric vehicles.

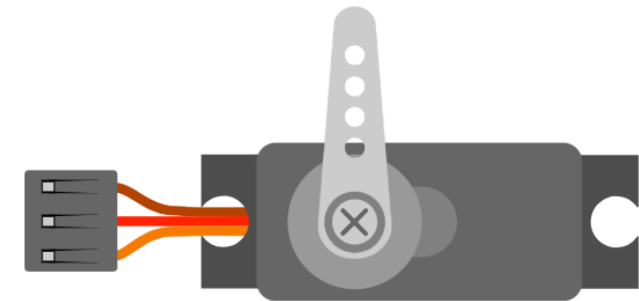
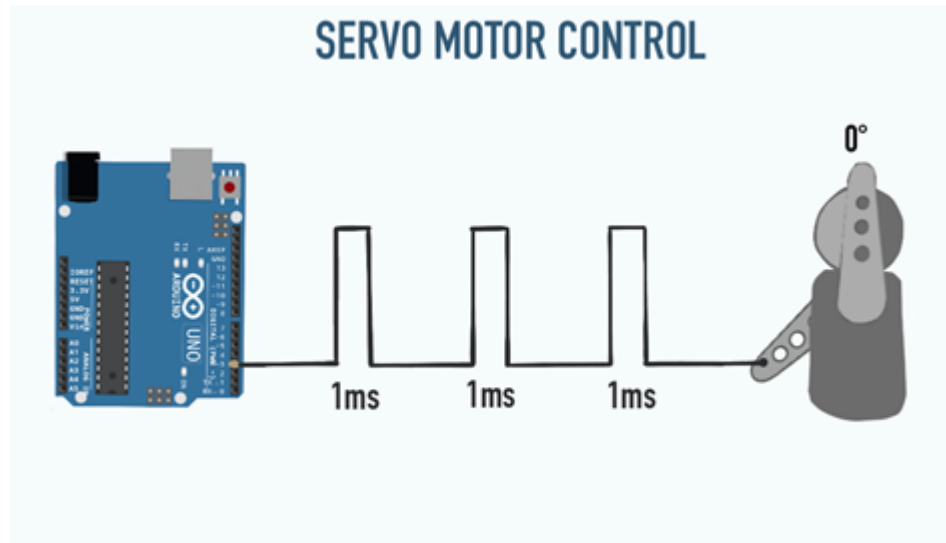
Chapter

02

Servo motor

You can rotate the servo motor shaft from 0° to 180° by varying the PWM pulse width from 1 ms to 2 ms.

PWM (Pulse Width Modulation)



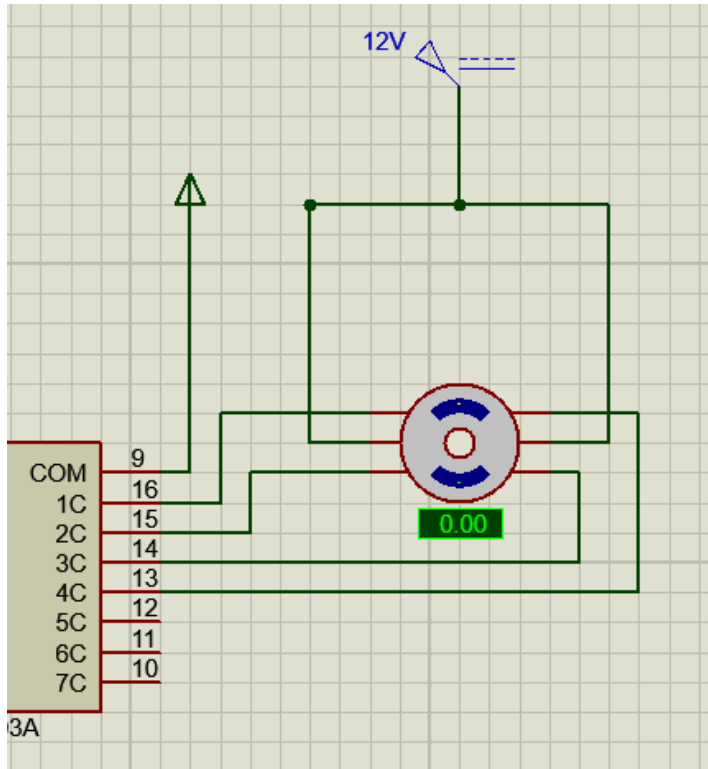
NAME	WIRE COLOR	DESCRIPTION
PWM	Yellow	Input control signal for the motor
V+	Red	Positive supply terminal
Ground	Brown	Ground terminal

Chapter

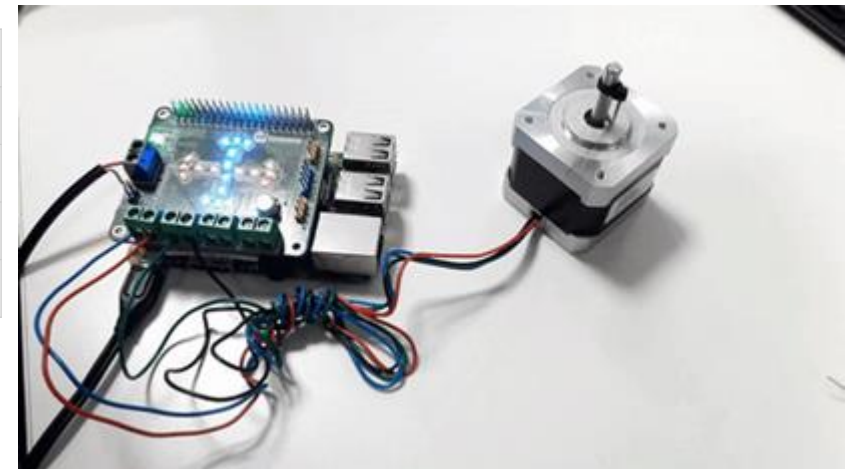
02

Stepper Motor

In the following table, A-B-C-D refers to the stator coils, that are to be energized sequentially in the manner and '1's and '0's refers to 'HIGH' and 'LOW' states.



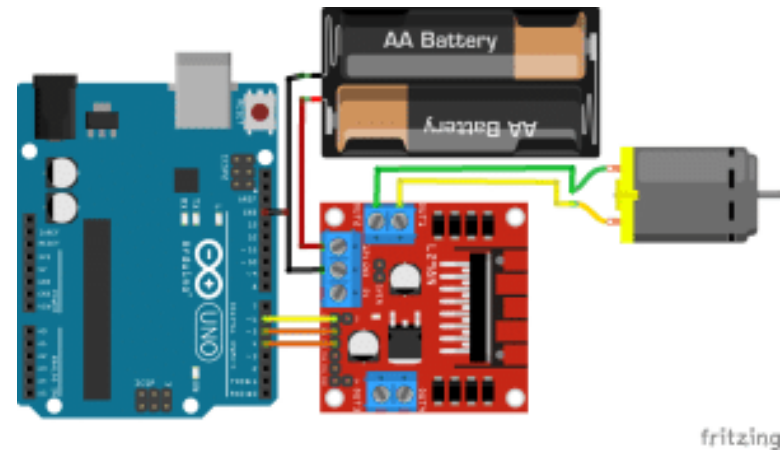
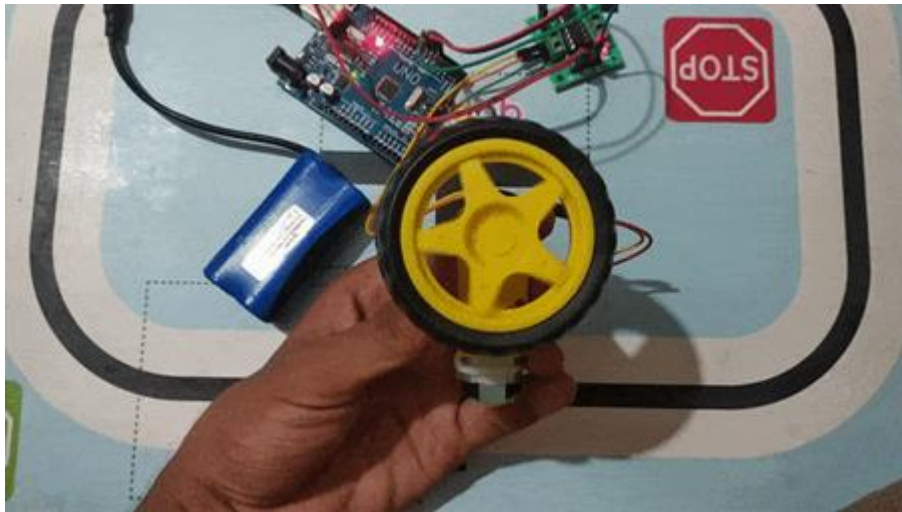
Steps	A	B	C	D	HEX
1	1	0	0	0	0x08
2	0	1	0	0	0x04
3	0	0	1	0	0x02
4	0	0	0	1	0x01



Chapter

02

DC Motor



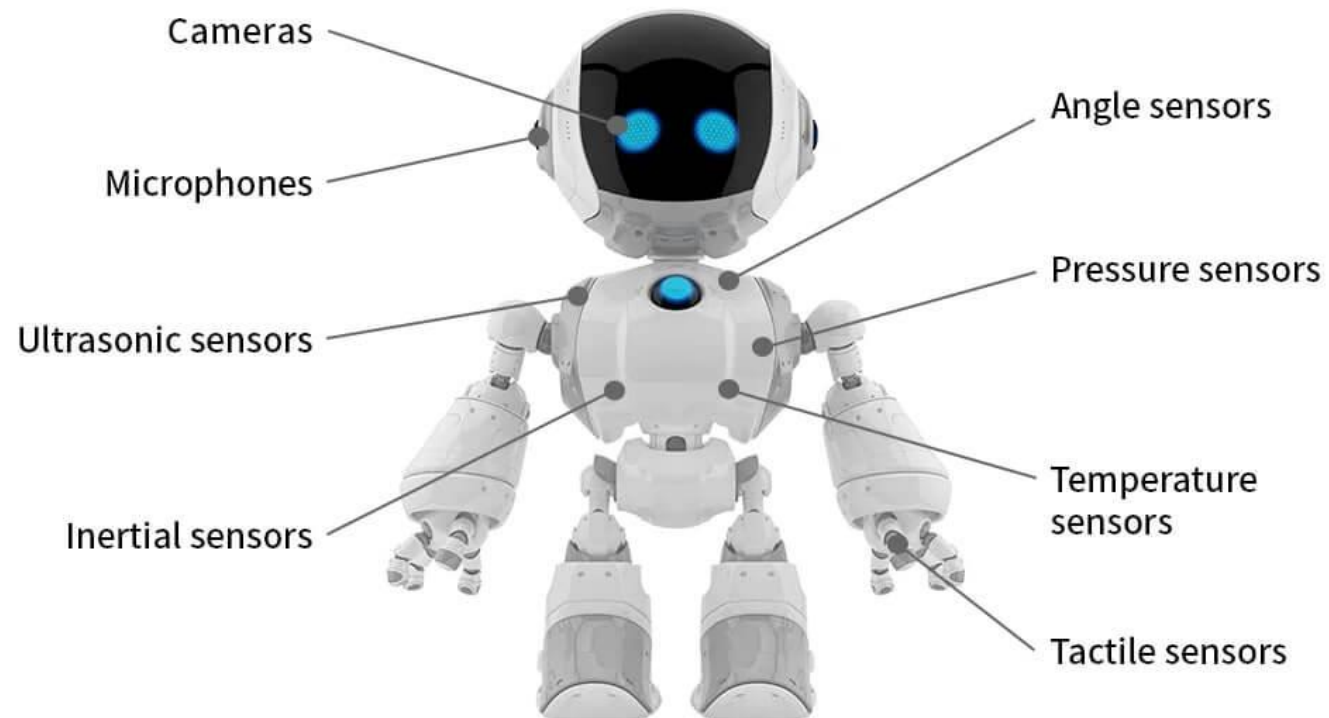
Arduino uno
D5
D4
GND

Motor driver
IN1
IN2
GND

fundamental differences between DC, Servo, and Stepper motors

Motor Type	Control	Precision	Speed	Torque	Primary Use in Robotics
DC Motor	Variable speed control	Low	High	Medium	Ideal for applications requiring variable speed and direction, such as robot wheels or drones.
Servo Motor	Positional control	High	Medium	High at low speeds	Best for precise control of movement, such as robotic arms or where specific positioning is crucial.
Stepper Motor	Step control	High	Low to Medium	High at low speeds	Suited for precise steps in motion, making them perfect for 3D printers, CNC machines, and robots requiring detailed positioning without feedback.

Sensors



Vision Sensors: The Eyes of Robots

• **Overview:** Introduction to how vision sensors allow robots to perceive their environment visually, simulate human sight.

• **Types:**

- **Cameras (2D and 3D):** Capture visual information. 2D cameras provide flat images, while 3D cameras add depth perception, crucial for tasks requiring spatial understanding.
- **LIDAR (Light Detection and Ranging):** Uses laser pulses to map out the robot's surroundings in high detail, essential for navigation and obstacle avoidance.

• **Applications:**

- Object recognition, navigation, and environmental mapping.



•**Overview:** Explaining the role of proximity and distance sensors in detecting the presence of objects and measuring the distance to them.

•**Types:**

- **Ultrasonic Sensors:** Emit ultrasonic waves and measure the reflection time to determine distance to objects. Widely used in robotic cars for obstacle detection.
- **Infrared Sensors:** Use infrared light to detect objects and estimate distance, useful in line-following robots and simple navigation tasks.

•**Applications:**

- Collision avoidance, area scanning, and precise movements in tight spaces.



The Ultrasonic Sensor

Feeling and Interacting: Tactile and Environmental Sensors

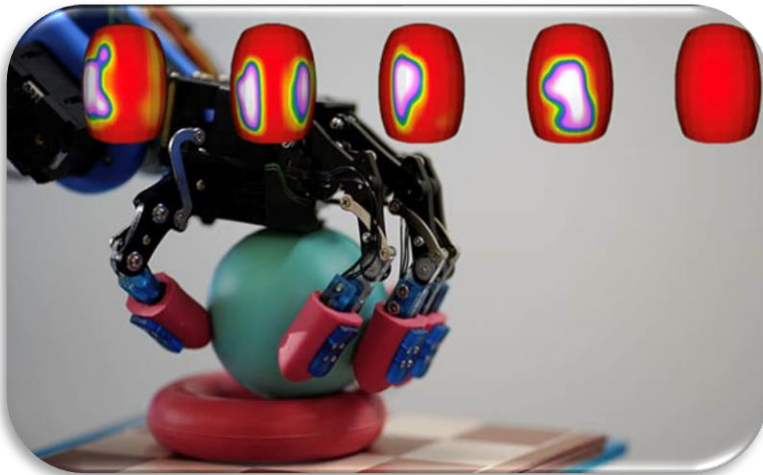
• **Overview:** Highlighting sensors that allow robots to 'feel' their environment and respond to physical interactions or changes in their surroundings.

• **Types:**

- **Tactile Sensors:** Enable robots to detect physical contact, pressure, and textures. Key in applications requiring delicate handling or manipulation, like robotic hands.
- **Environmental Sensors:** Measure temperature, humidity, or gas concentrations, critical for robots operating in varying or extreme conditions, such as disaster response robots.

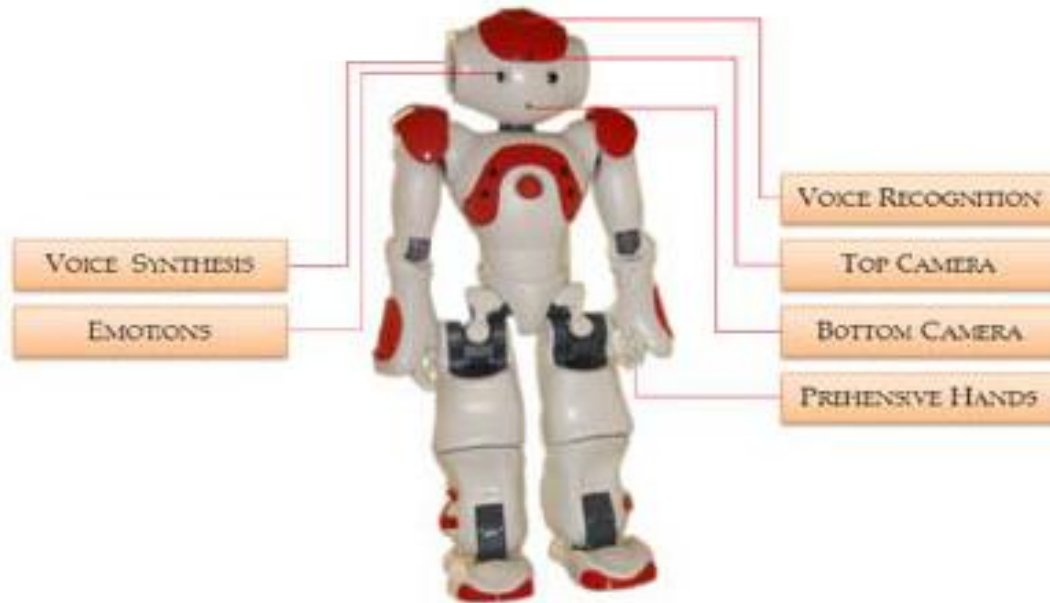
• **Applications:**

- Hazard detection, material handling, and adapting to environmental conditions.

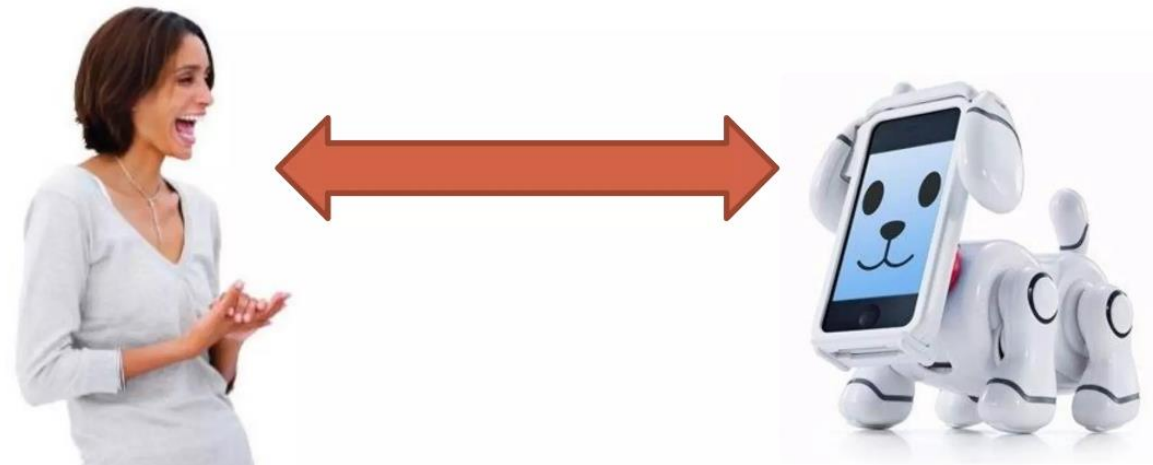


Introduction to Human-Robot Interaction (HRI)

- Brief introduction to HRI as the study of how humans and robots interact and communicate.
- Highlight the importance of HRI in integrating robots into human environments, from homes to workplaces.



speech, text, touch screen, image, video, gestures, emotions, sound localization, voice biometrics, face recognition...



Chapter

03

Week 3 Outlines



Course Outlines

Module	Content	Simulation Lab
1. Introduction to Robotics	- Fundamentals of Robot Control - Classification of Robotics - Exploration of Robot Types	- Basic Robot Control - Modeling Robot Classes - Simulating Various Robot Types
2. Advanced HRI Studies	- Introductory to Advanced Concepts in HRI	- Basic to Complex HRI Scenarios
3. Social Dynamics in Robotics	- Architectures for Social Interaction - Case Studies and History of HRI	- Social Interaction Simulations - Historical Evolution in HRI



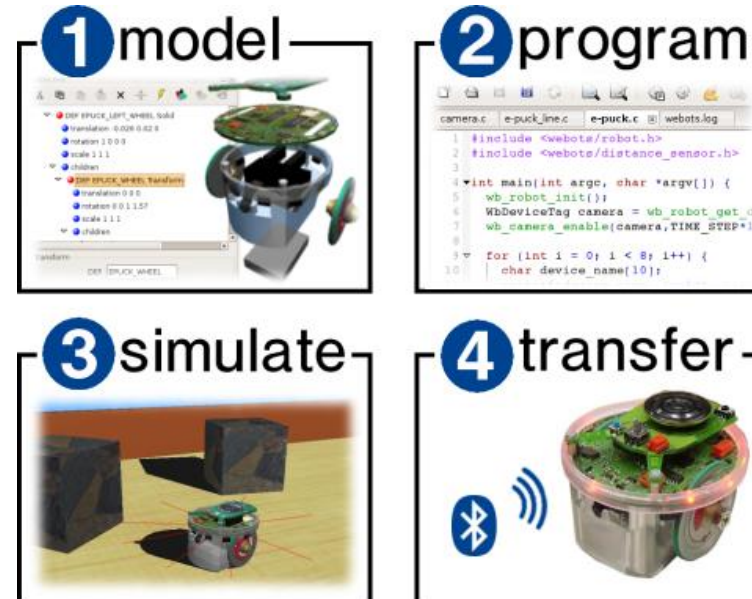
Webots
robot simulation

Webots is an open source and multi-platform desktop application used to **simulate** robots

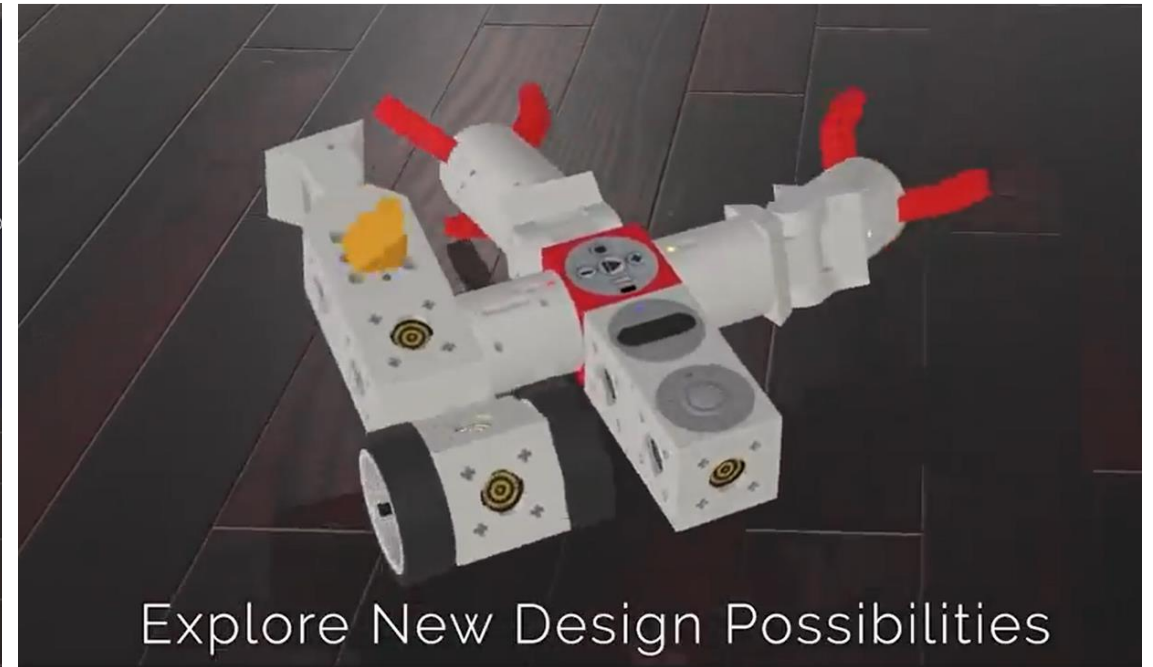
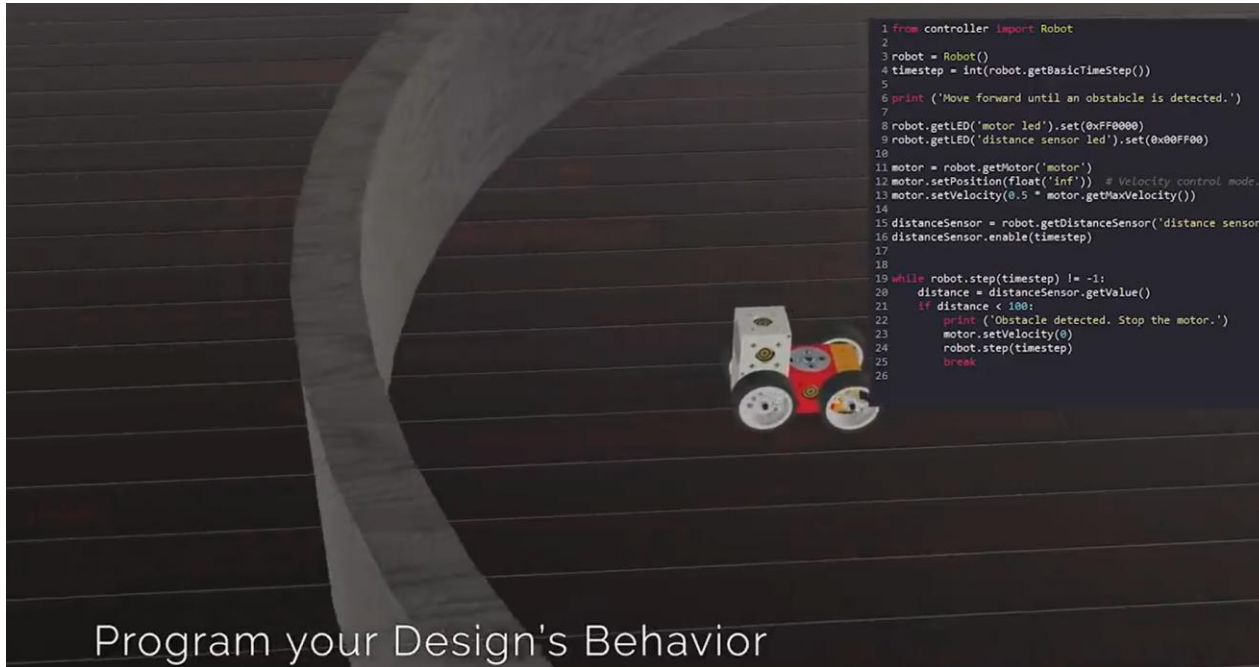
<https://cyberbotics.com/>

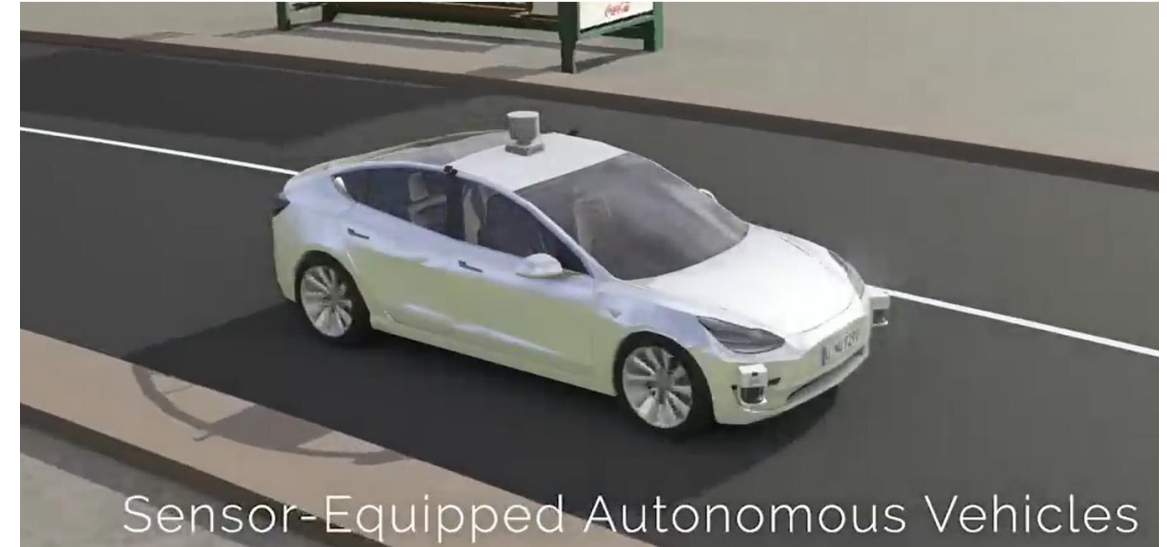


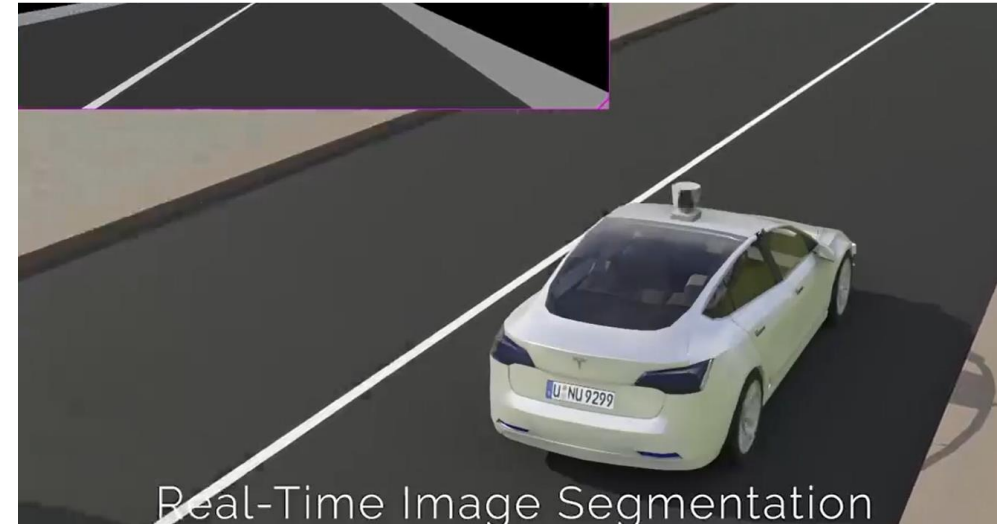
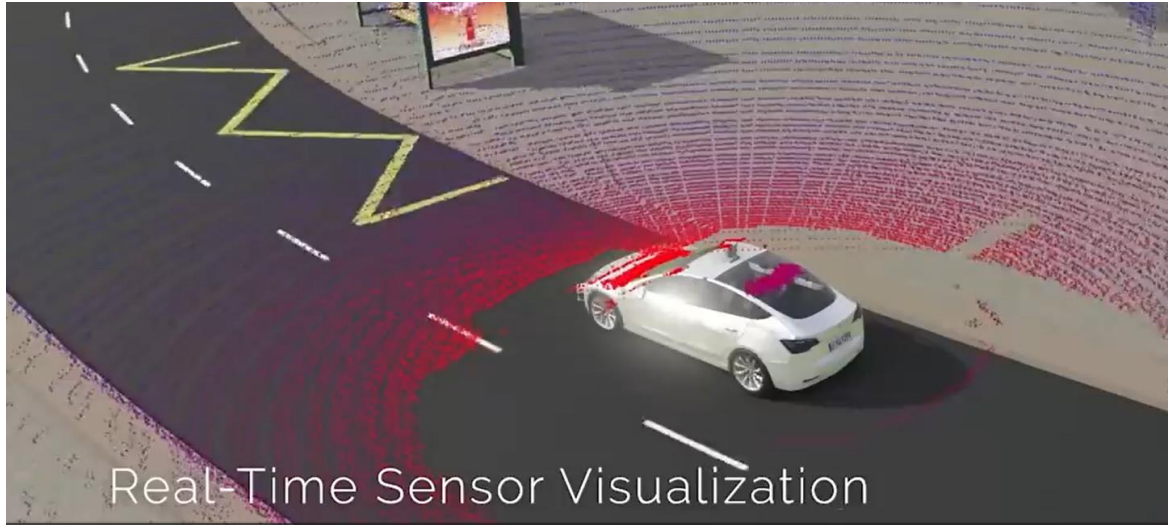
Webots Open Source











Webots is Amazing Because:

- **Real Robots on Your Screen:** Your virtual robots in Webots move just like real ones. It's like having a robot pal in your computer!
- **Create Your Own Robot Places:** Make all sorts of places for your robots to explore, like cities, parks, or space stations!
- **Loads of Ready-Made Robots:** Choose from many robots that are already built in Webots. Start playing and learning right away.
- **Works on Any Computer:** You can use Webots on Windows, Mac, or Linux. It's super flexible!
- **Easy Robot Coding with Python:** Use Python, a simple programming language, to tell your robots what to do. Perfect for beginners and experts!
- **Teach Robots to Think:** Make your robots smart with Python. They can learn to solve problems by themselves, like real AI!
- **Robots with Cool Parts:** Add neat parts to your robots, like cameras, hands, or sensors, so they can see, touch, and feel.
- **All-in-One Robot Kit:** Webots has everything you need to create and control amazing robots.
- **For School, Research, and Business:** You can play with Webots alone, use it in school, or even in big research projects and businesses.
- **Used by Top Universities and Companies:** Many of the best universities and big companies in America use Webots. Places like MIT, Stanford, and tech giants in Silicon Valley trust Webots for their robot projects.

Download & Install
User interface (UI)
Create your first project
Make your own object
Understand fundamentals
Run your first simulation
Add a robot & run simulation
Where is the code

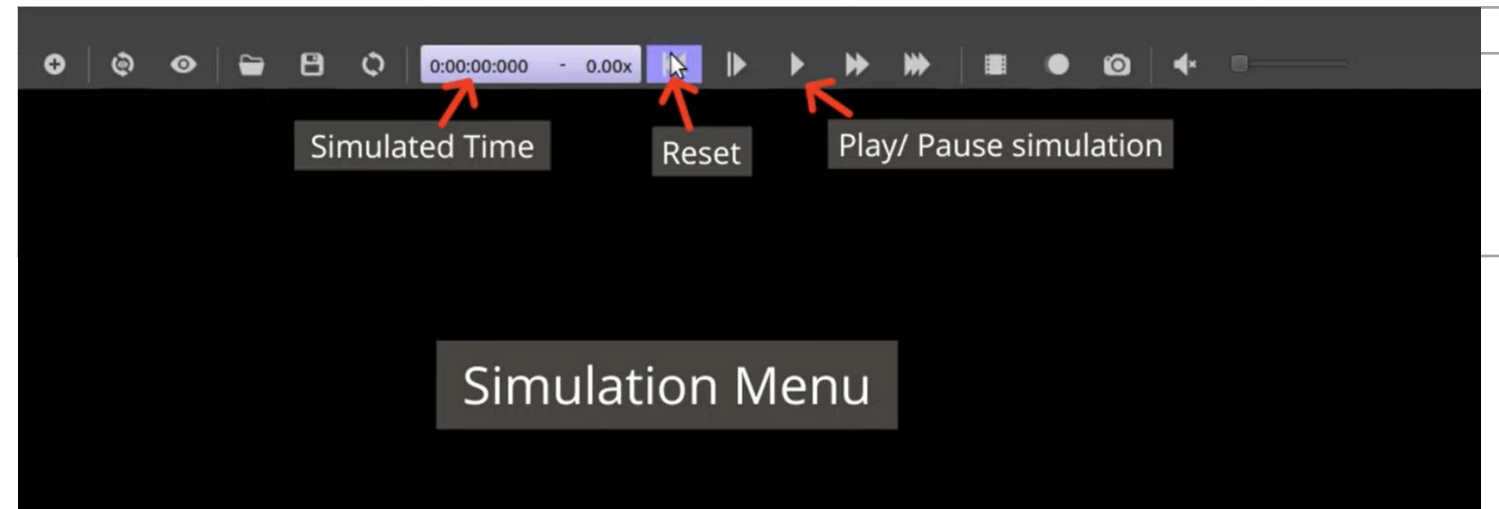
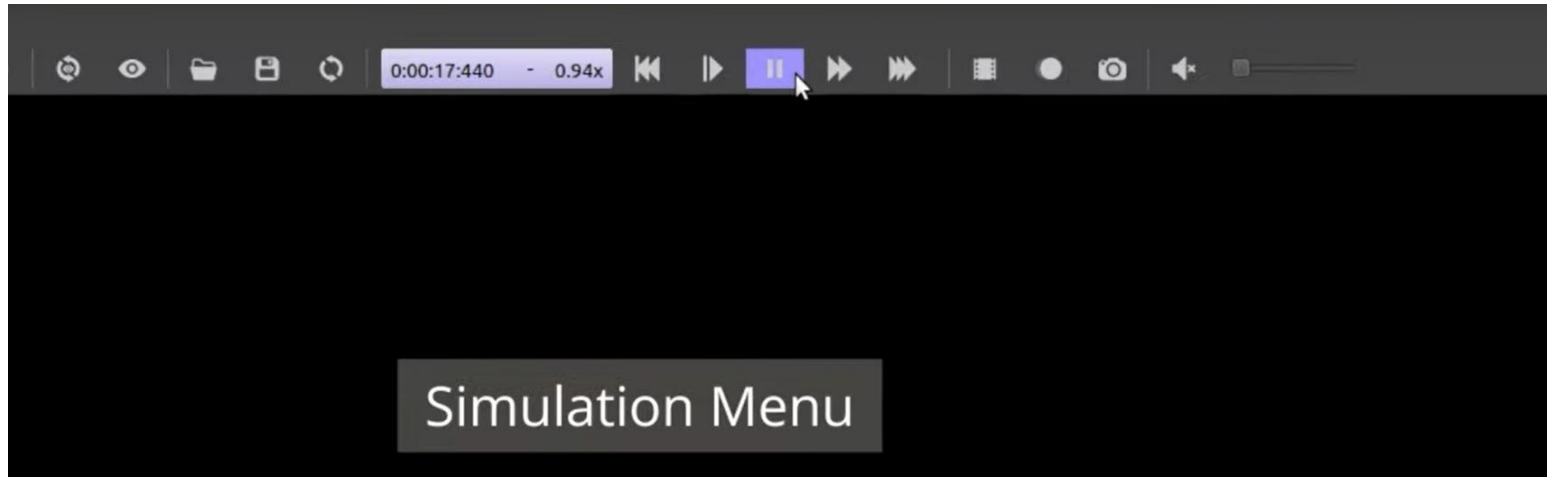
Objective

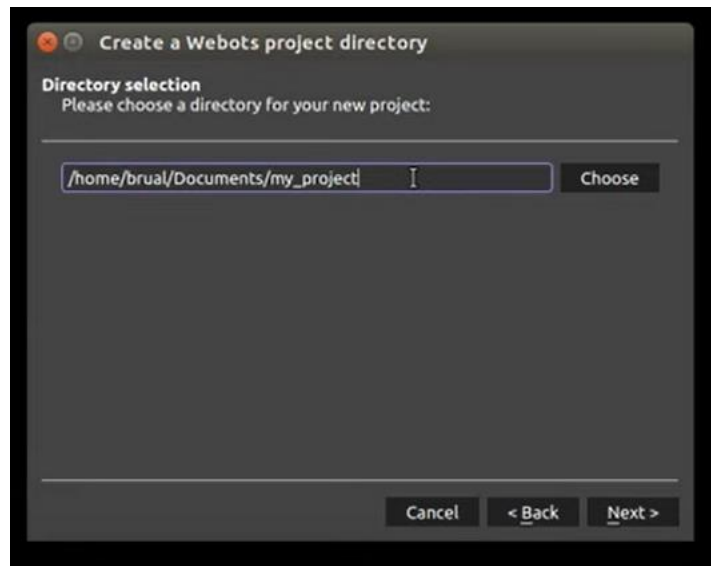
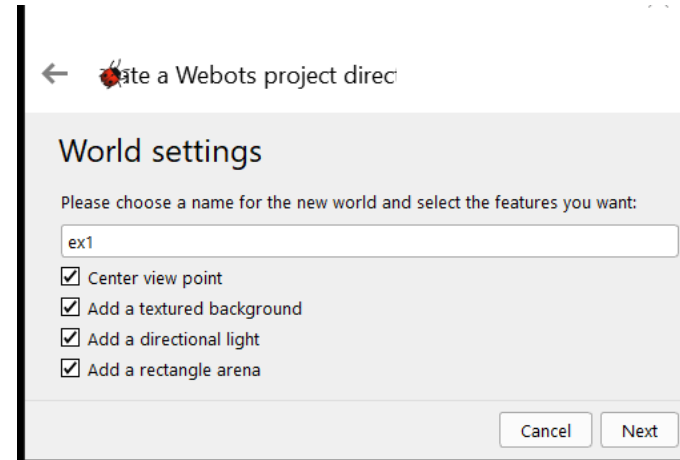
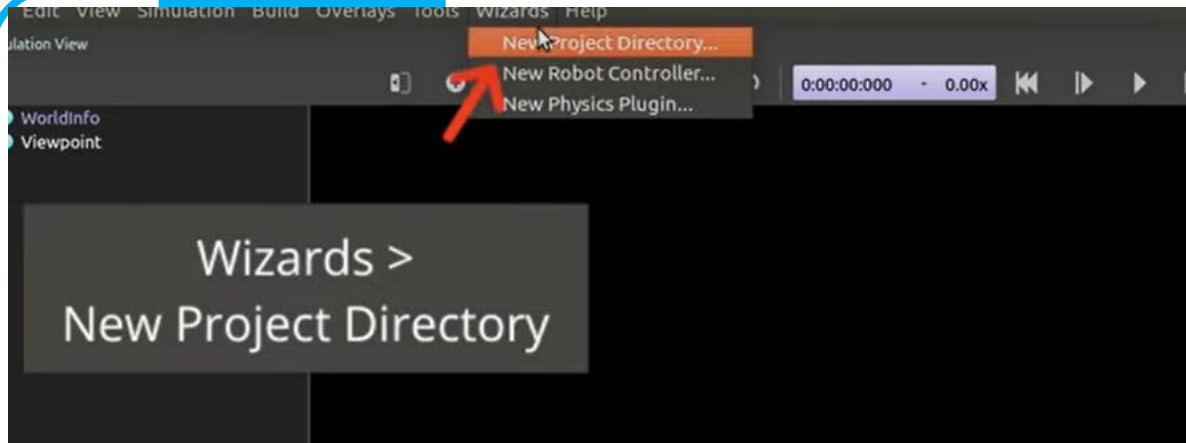
User Interface (UI)

Create your first Project

Chapter 03

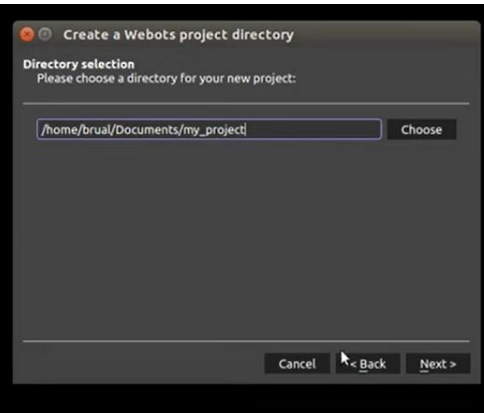
User Interface (UI)





(1) Use "Choose" to
choose where to save
your project.

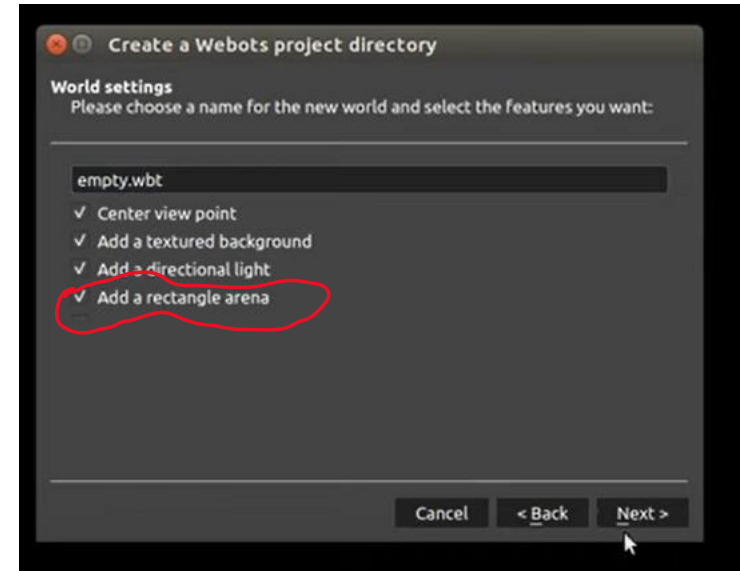
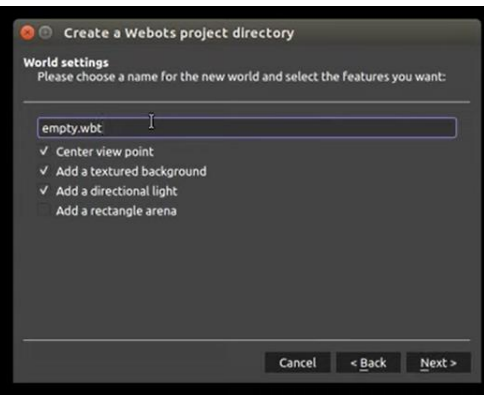
(2) Name your project.
Eg: my_project

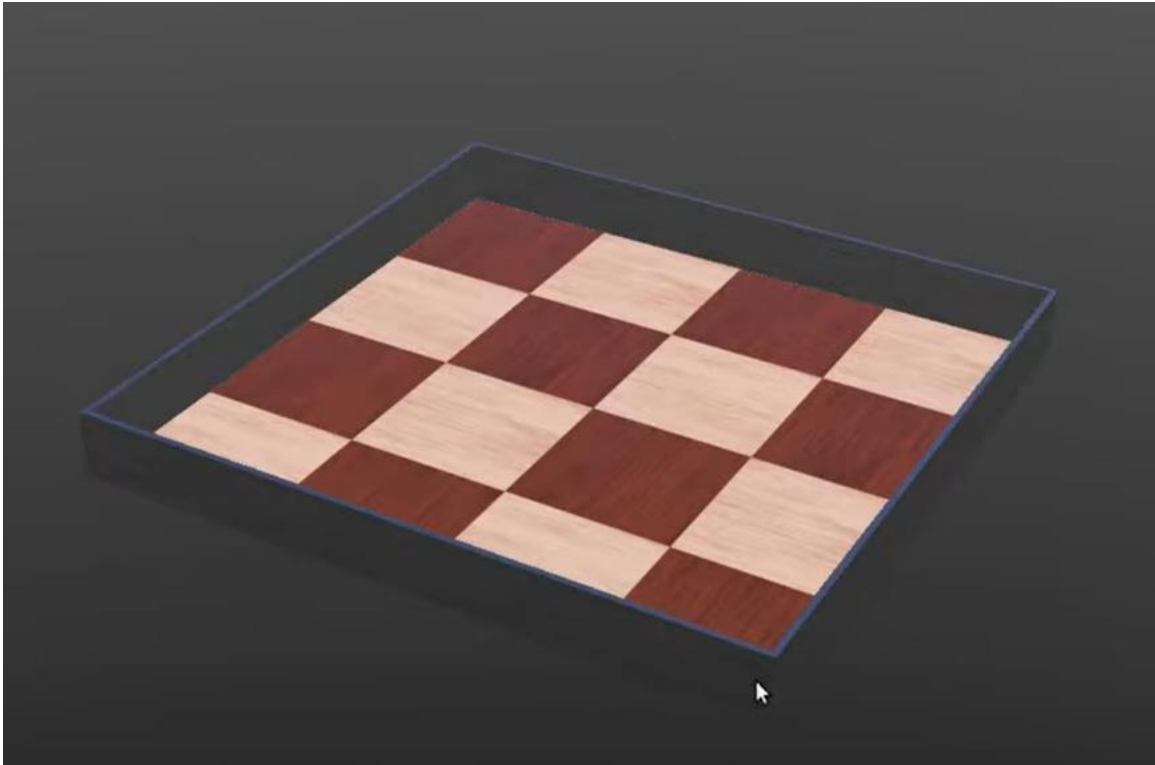


(3) Click Next

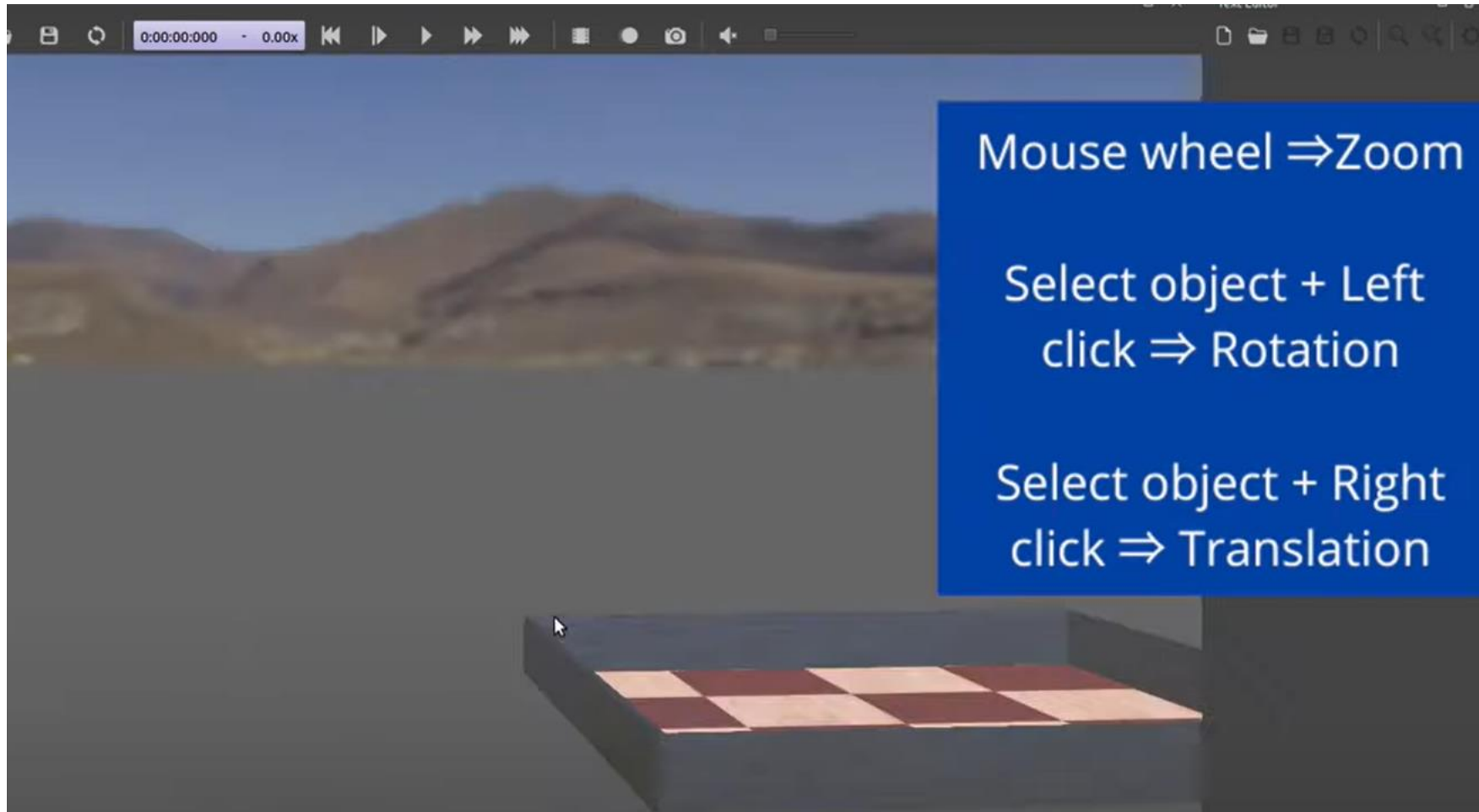
(4) Select "Add
rectangle arena"

(5) Click Next & Finish

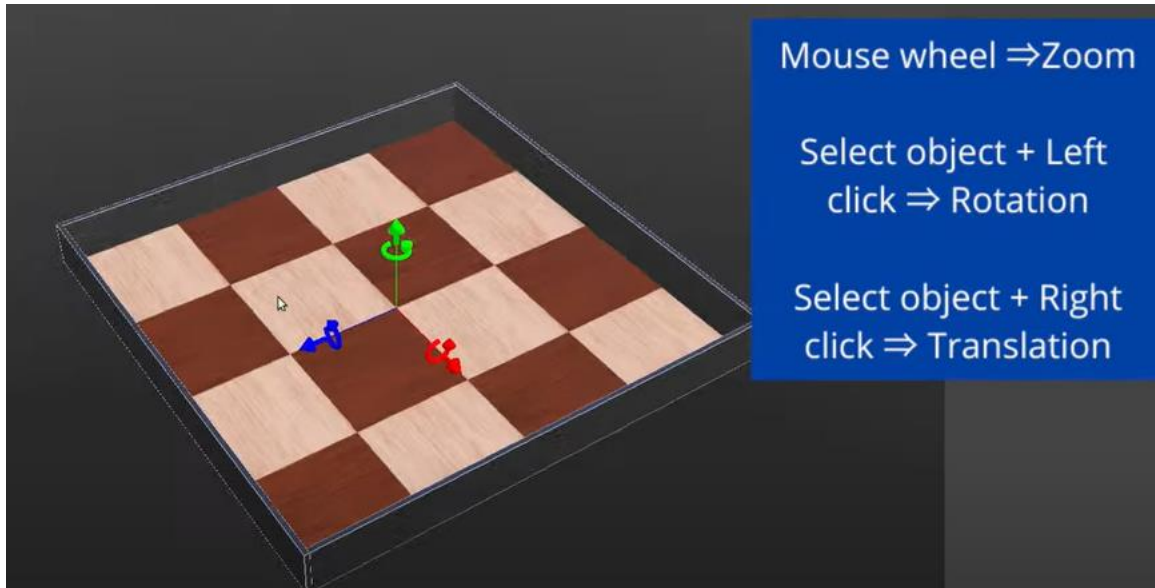




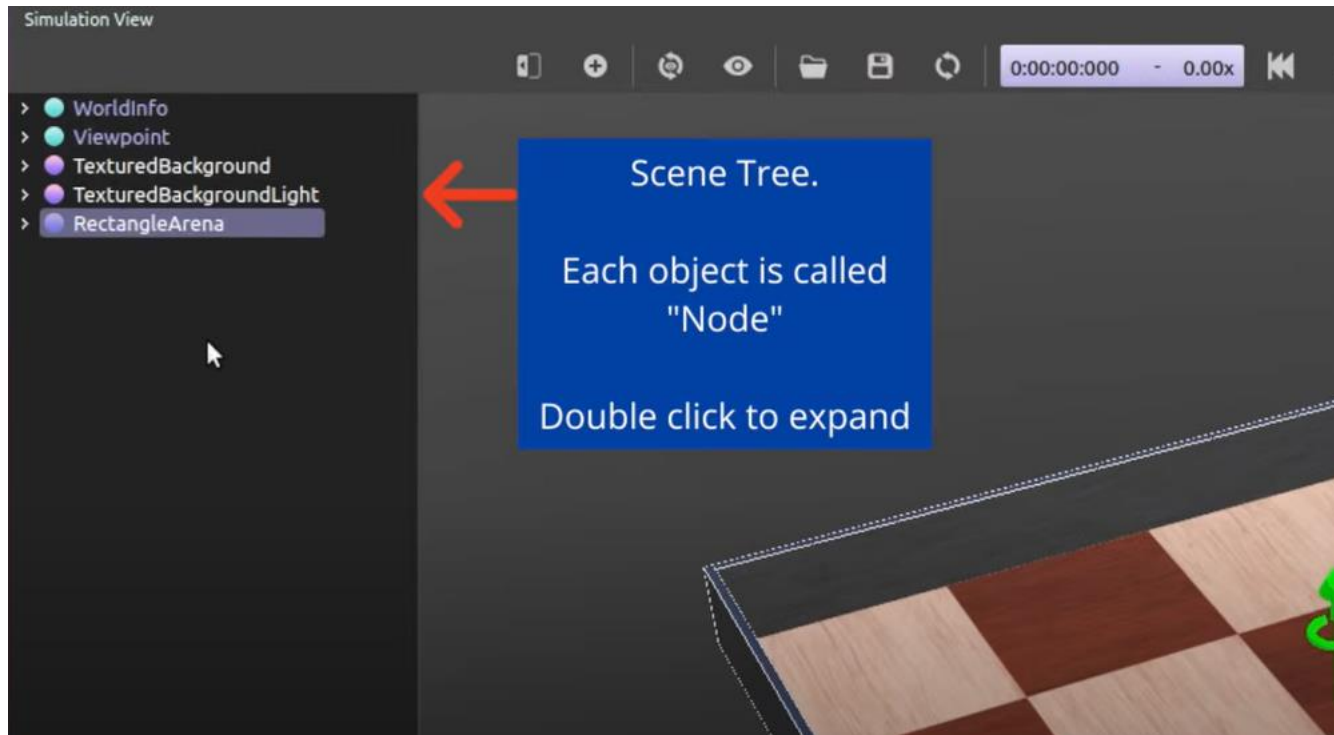
Loaded with a rectangular arena
essentially a floor with four walls.



learn how to zoom in and zoom out this is simply done by scrolling in and out

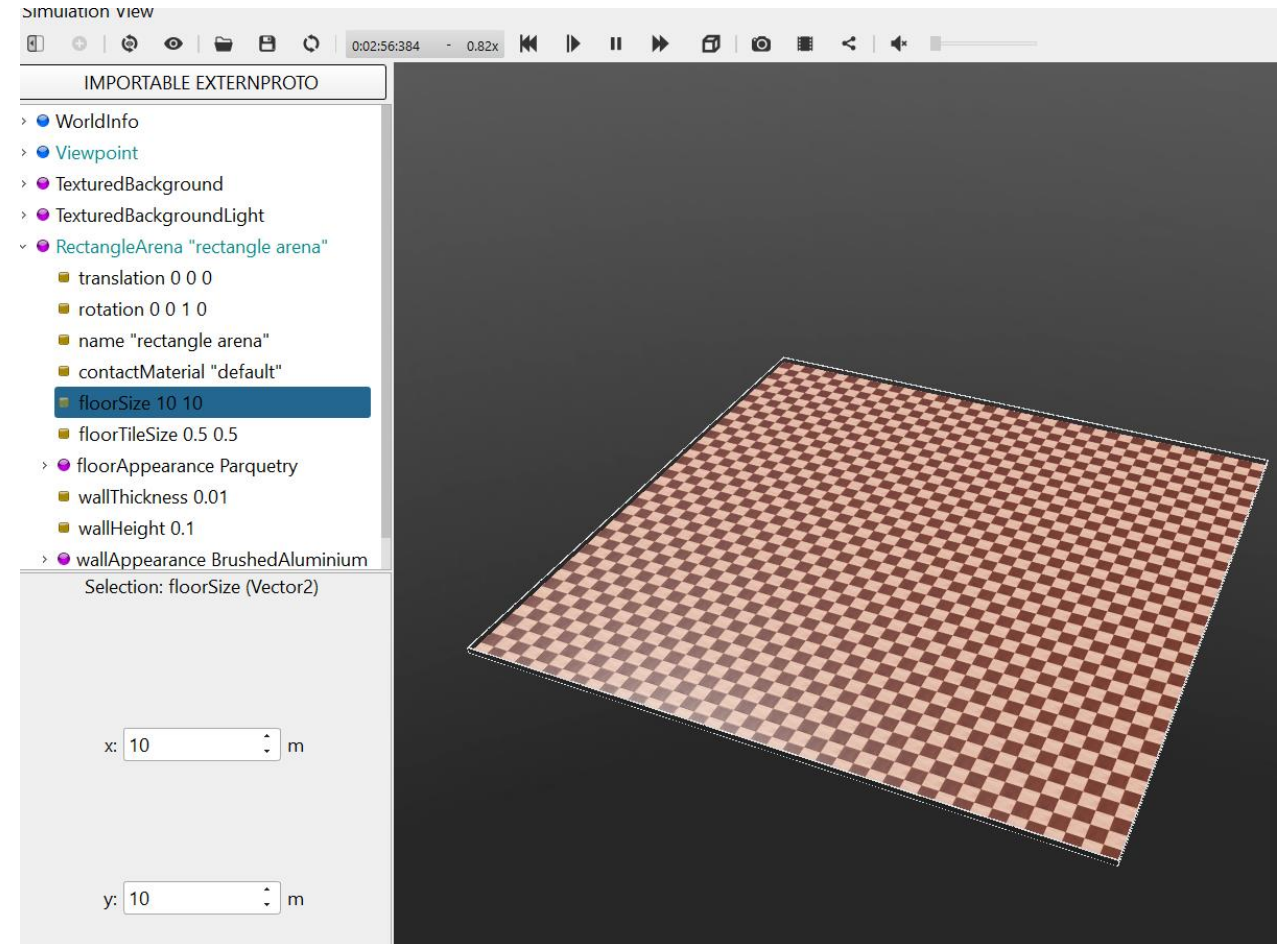
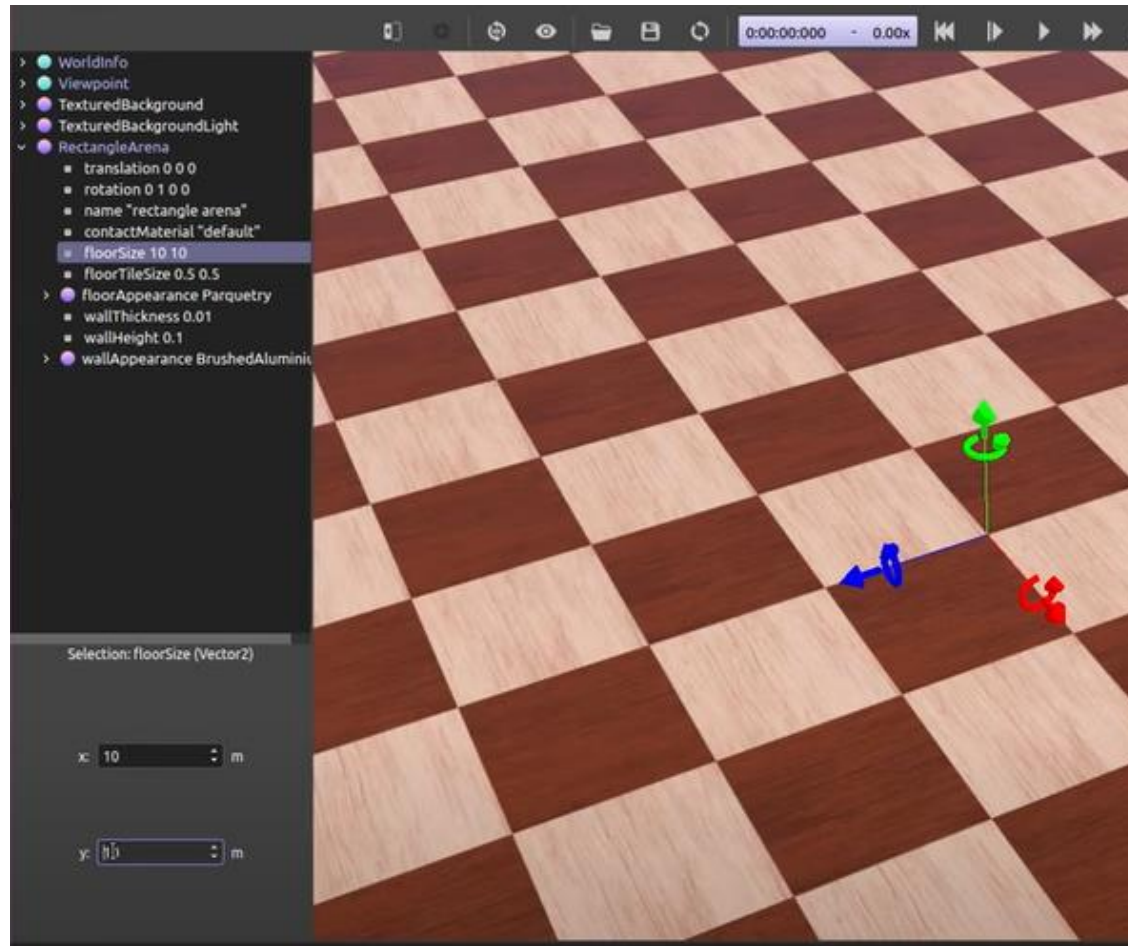


you can click on it and you can move it around to look at different views.

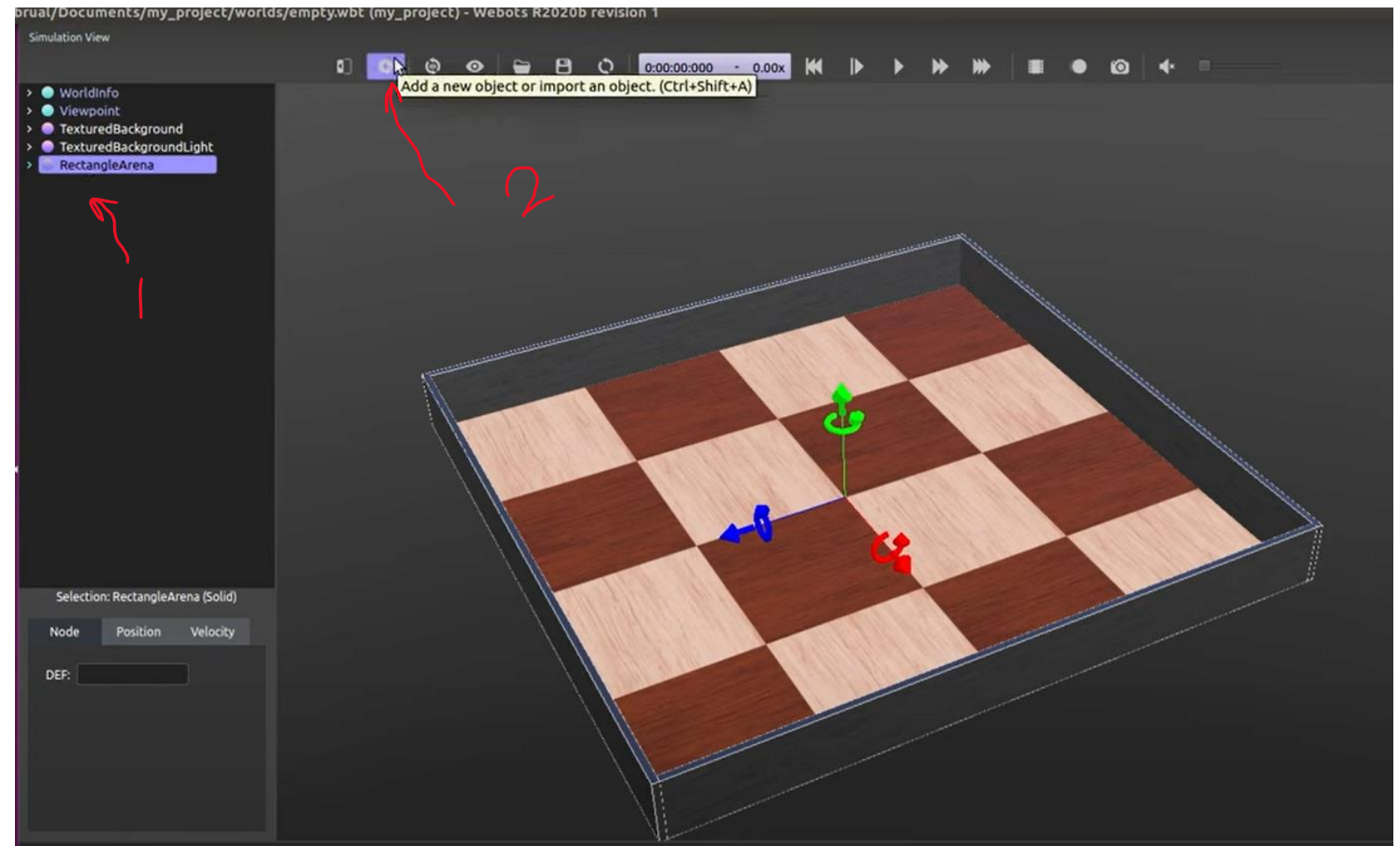


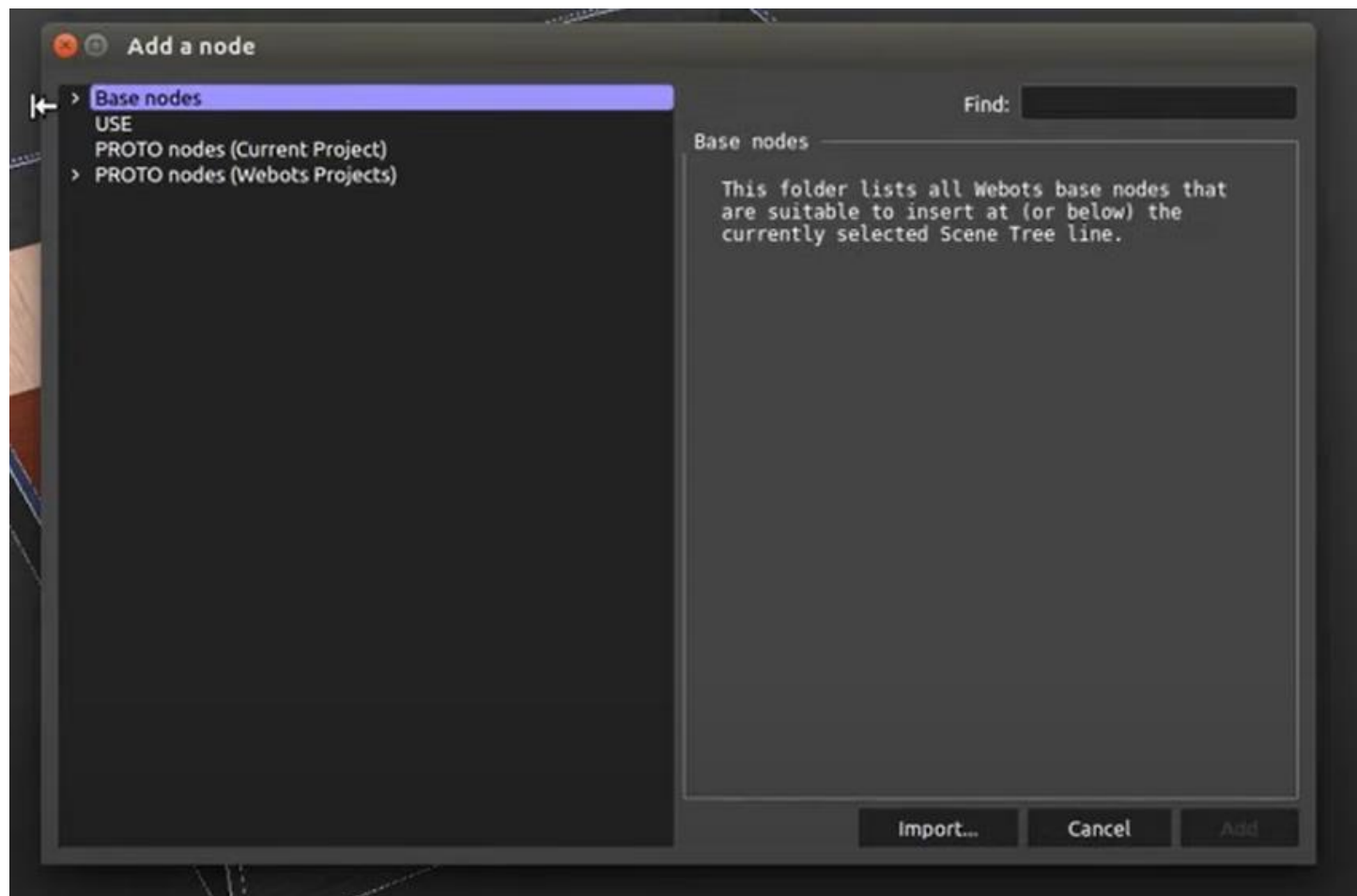
Chapter 03

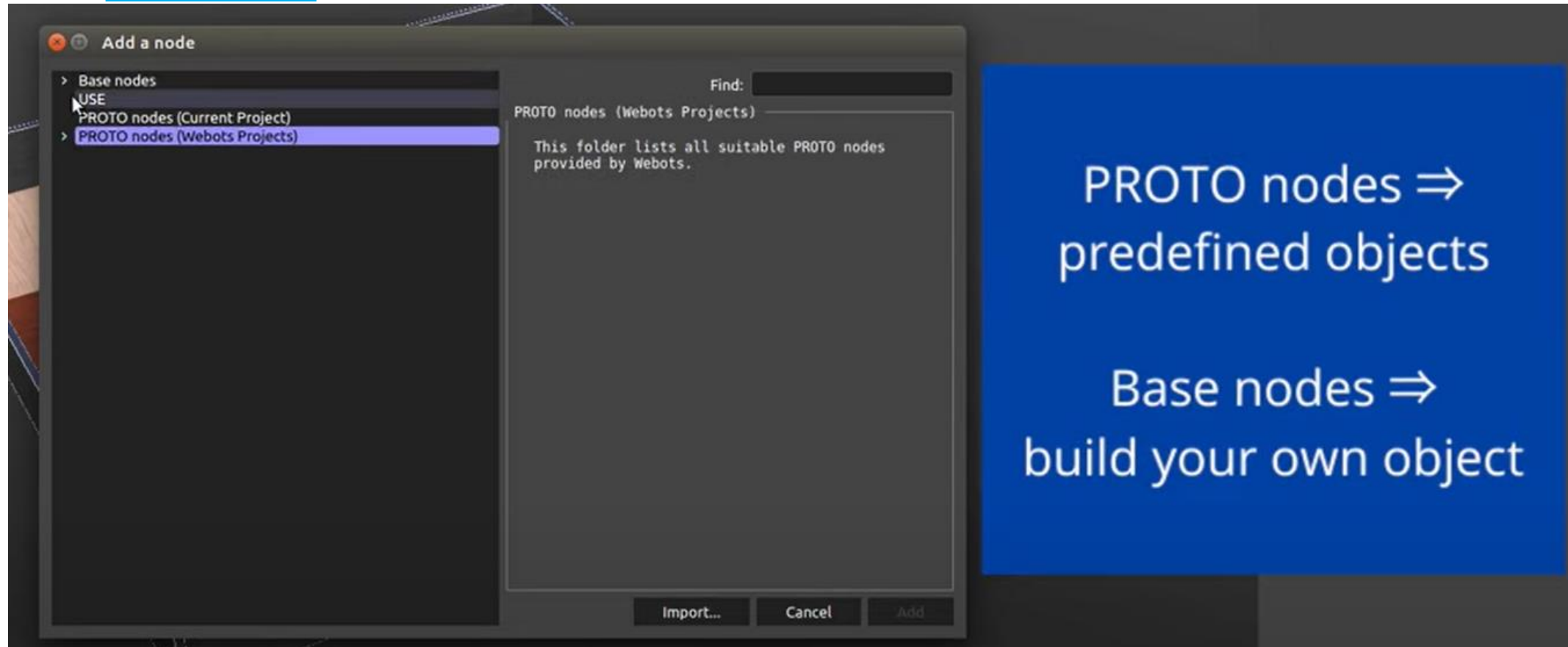
Change the Floor Size

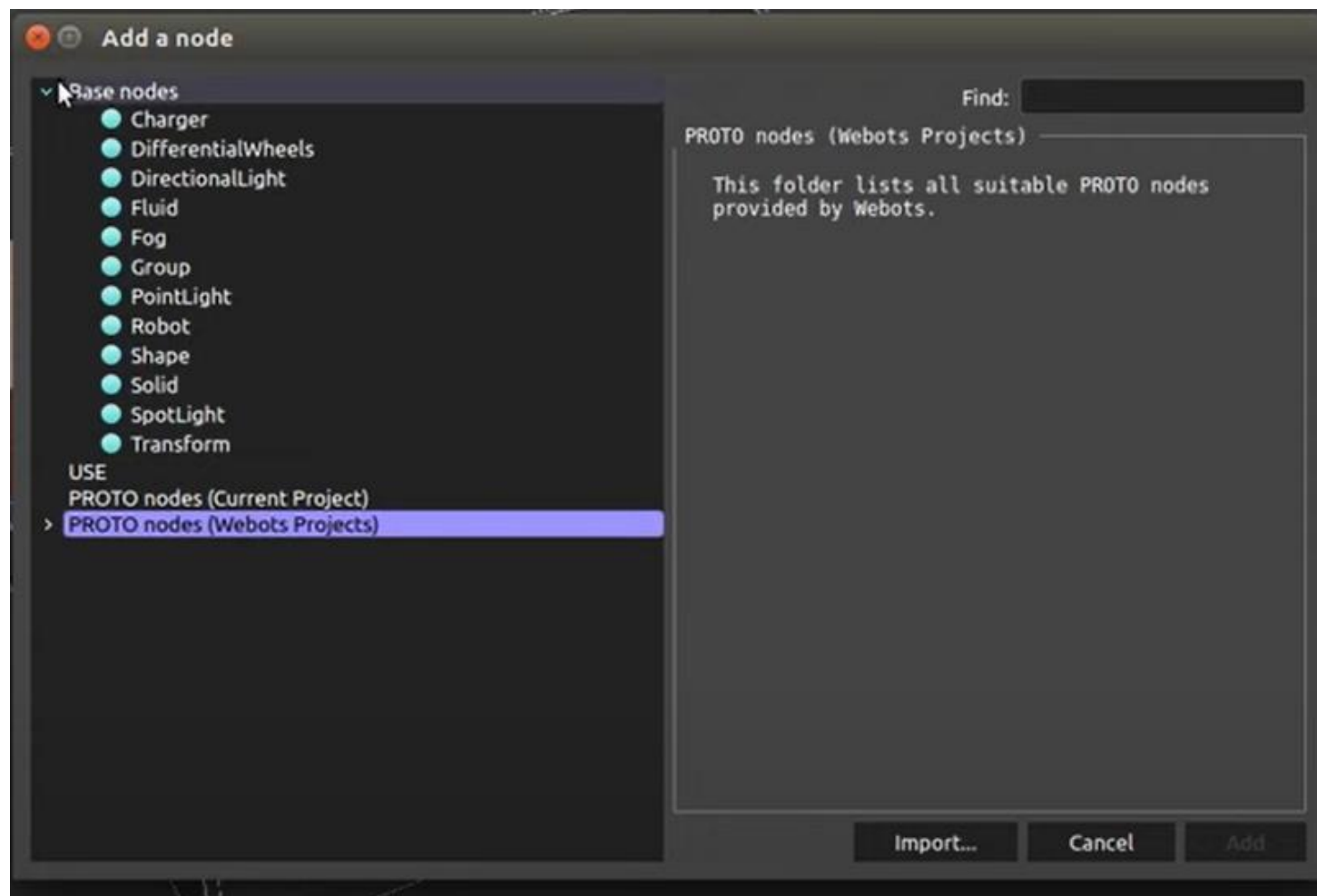


create an object of our own we will first go over the steps to create an object

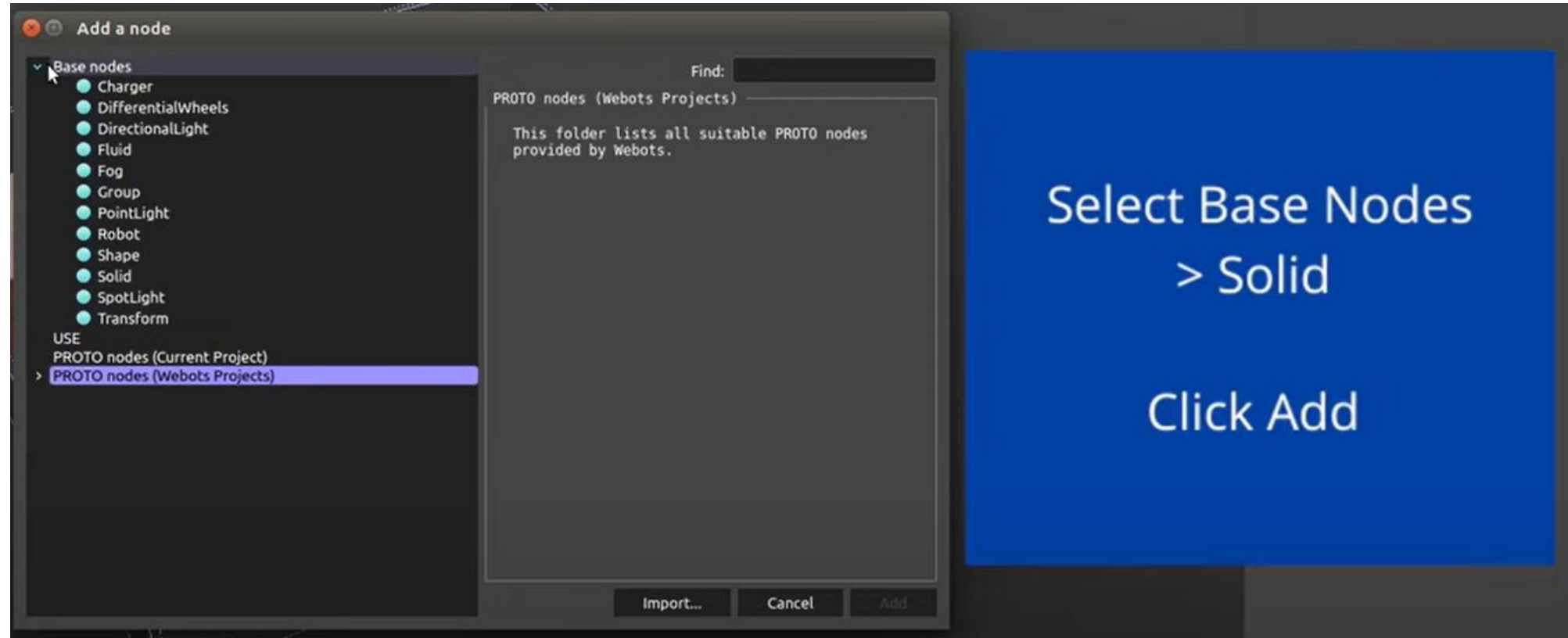


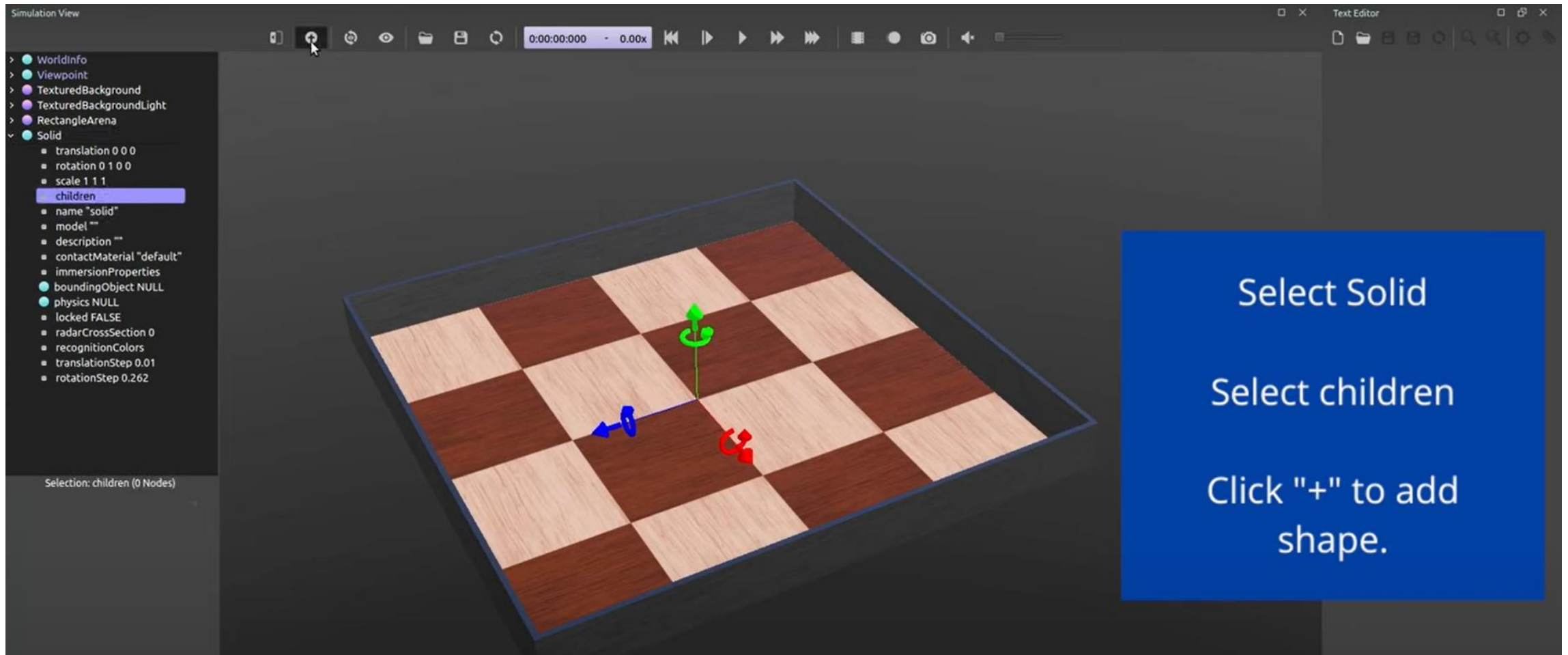


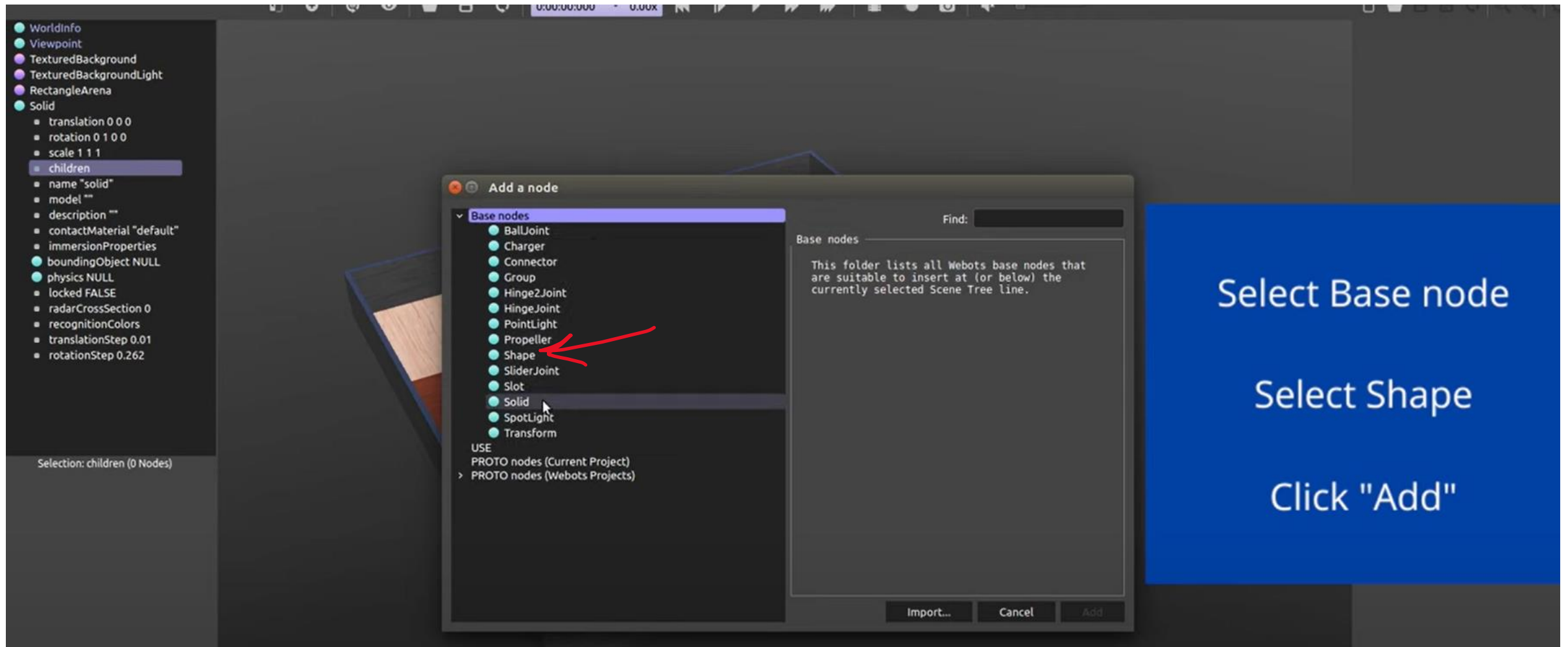


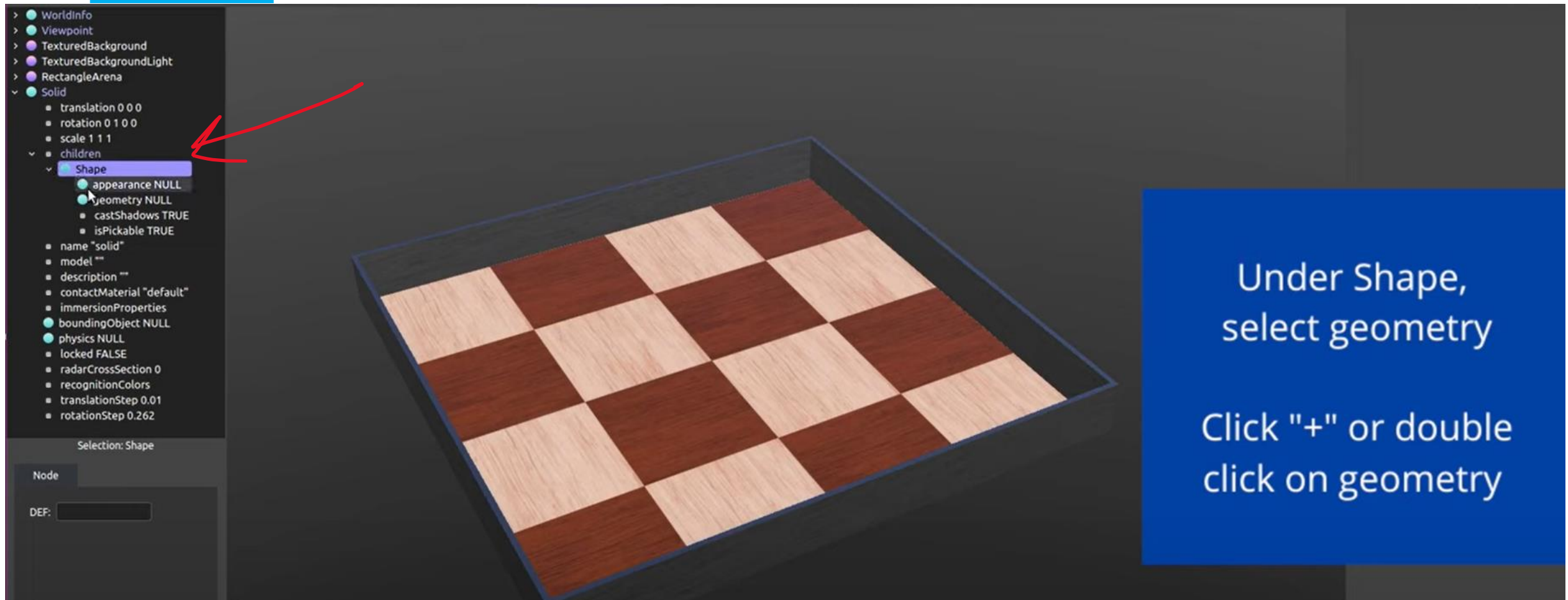


Create a NEW Object:









Chapter 03

Simulation View

WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
RectangleArena
Solid
 translation 0 0 0
 rotation 0 1 0
 scale 1 1 1
 children
 Shape
 appearance NULL
 geometry NULL
 castShadows TRUE
 isPickable TRUE
 name "solid"
 model ""
 description ""
 contactMaterial "default"
 immersionProperties
 boundingObject NULL
 physics NULL
 locked FALSE
 radarCrossSection 0
 recognitionColors
 translationStep 0.01
 rotationStep 0.262

Selection: geometry

Node

Add a node

Base nodes

- Box
- Capsule
- Cone
- Cylinder
- ElevationGrid
- IndexedFaceSet
- IndexedLineSet
- Mesh
- Plane
- PointSet
- Sphere

USE

PROTO nodes (Current Project)

PROTO nodes (Webots Projects)

- objects
 - geometries
 - Extrusion (IndexedFaceSet)
 - Rectangle (IndexedFaceSet)
 - TexturedBox (IndexedFaceSet)
 - TexturedParallelepiped (IndexedFaceSet)
 - traffic
 - TrafficLightArrowLampGeometry (IndexedFa...
 - TrafficLightStandardLampGeometry (Sphere)
 - samples
 - geometries
 - TeapotMesh (IndexedFaceSet)

Base nodes

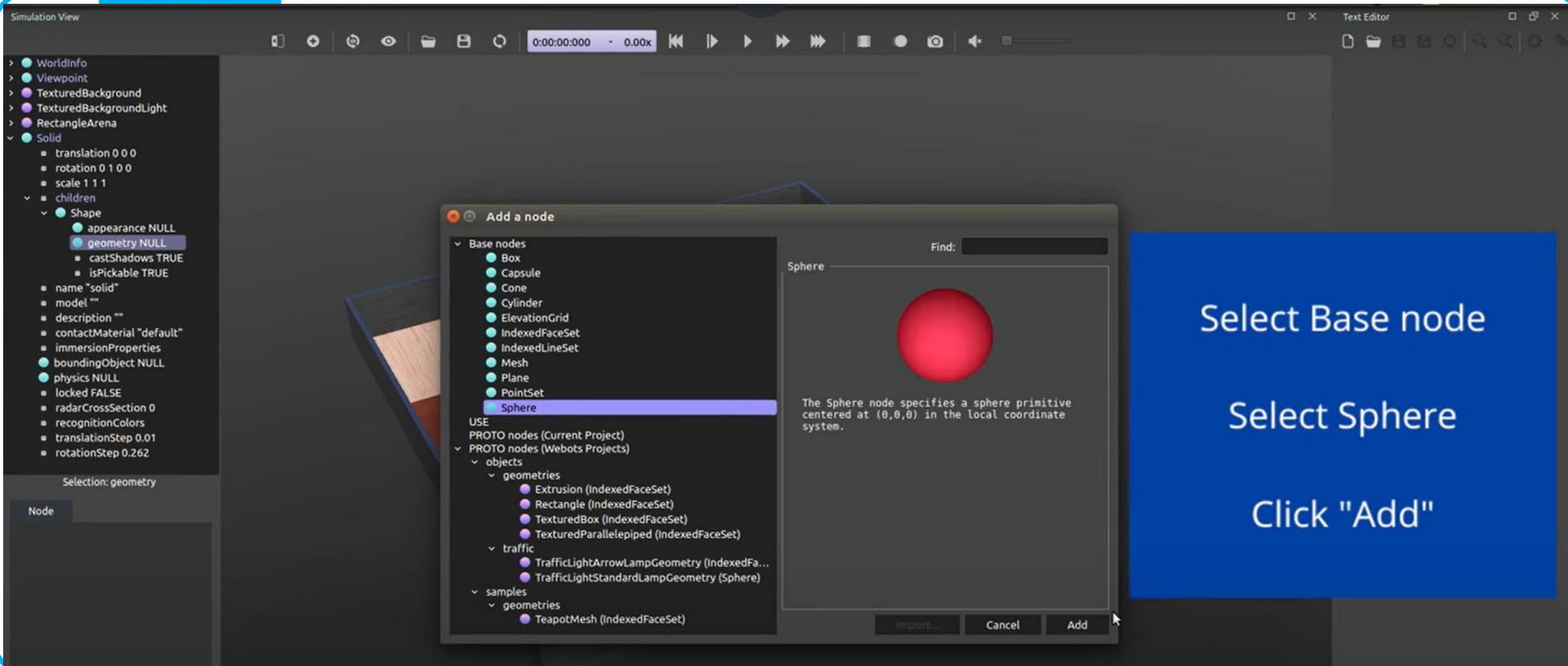
This folder lists all Webots base nodes that are suitable to insert at (or below) the currently selected Scene Tree line.

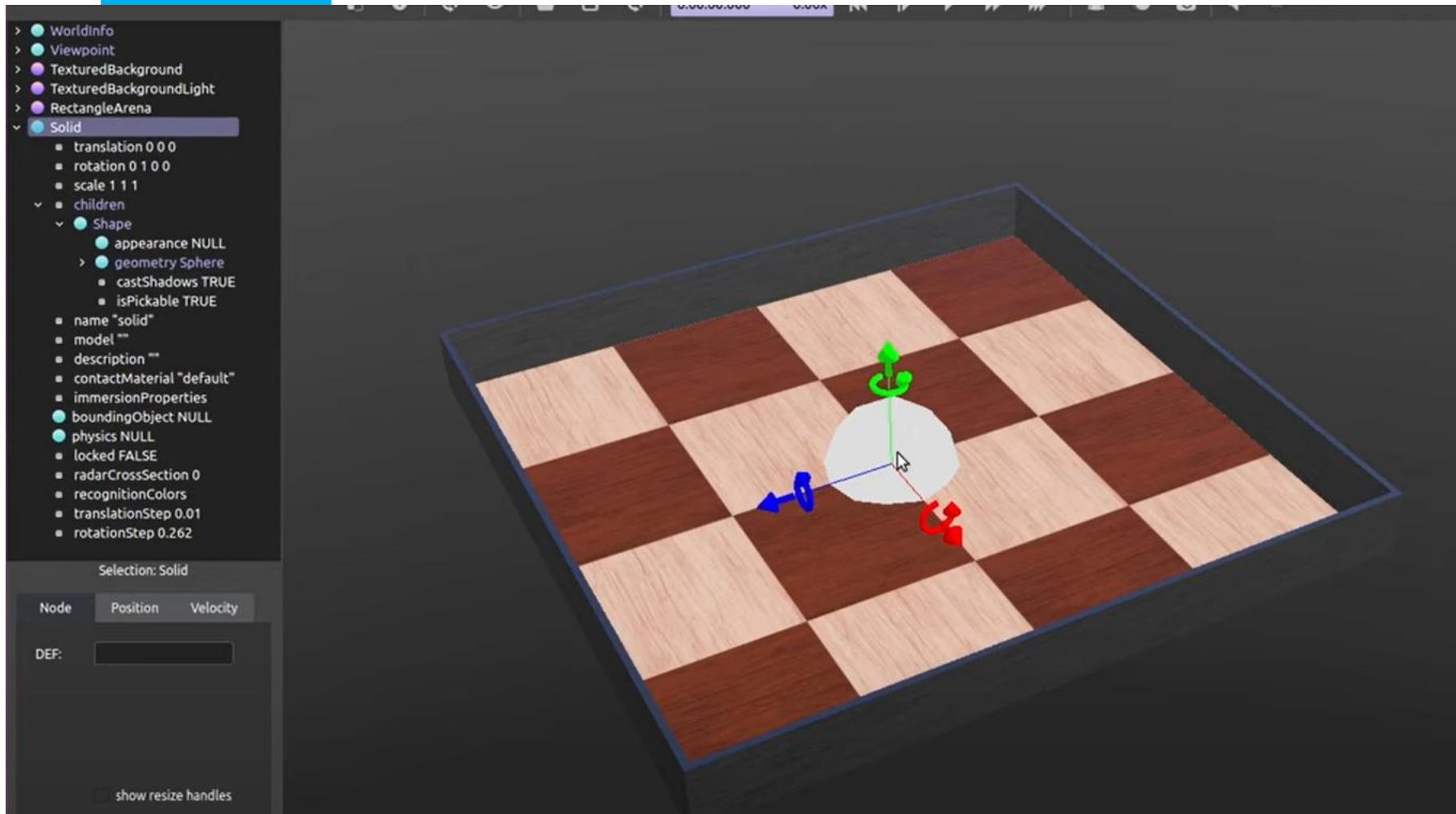
Find:

Insert Cancel Add

Under Shape, select geometry

Click "+" or double click on geometry





All objects added at (0,0,0).

- > WorldInfo
- > Viewpoint
- > TexturedBackground
- > TexturedBackgroundLight
- > RectangleArena
- > Solid
 - translation 0 0 0
 - rotation 0 1 0 0
 - scale 1 1 1
 - children
 - Shape
 - appearance NULL
 - geometry Sphere
 - radius 0.1
 - subdivision 1
 - ico TRUE
 - castShadows TRUE
 - isPickable TRUE
 - name "solid"
 - model ""
 - description ""
 - contactMaterial "default"
 - immersionProperties
 - boundingObject NULL
 - physics NULL
 - locked FALSE
 - radarCrossSection 0
 - recognitionColors
 - Selection: radius (Float)

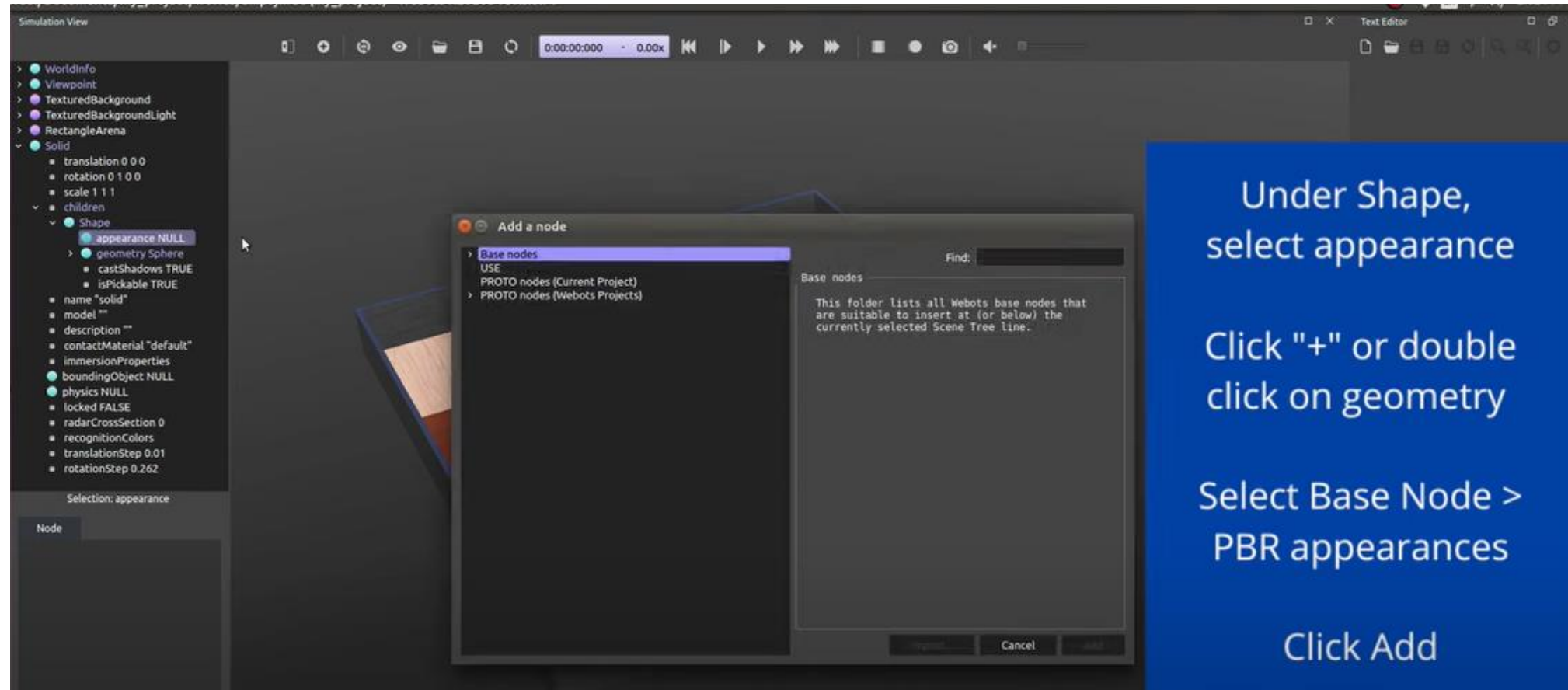
Under geometry,
there are options to
change size.

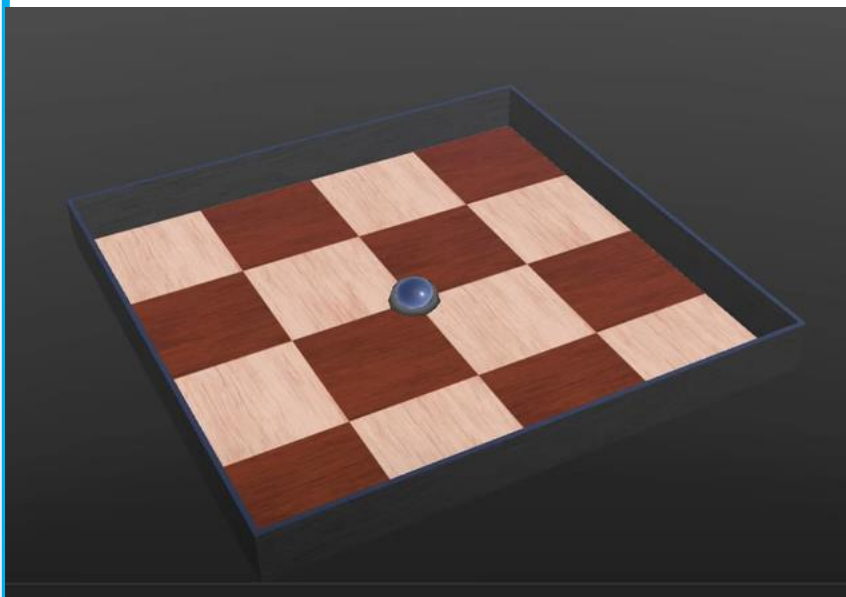
Select radius

the dimensions so we could go
from 0.1 to
0.05 and you can see the
sphere has reduced

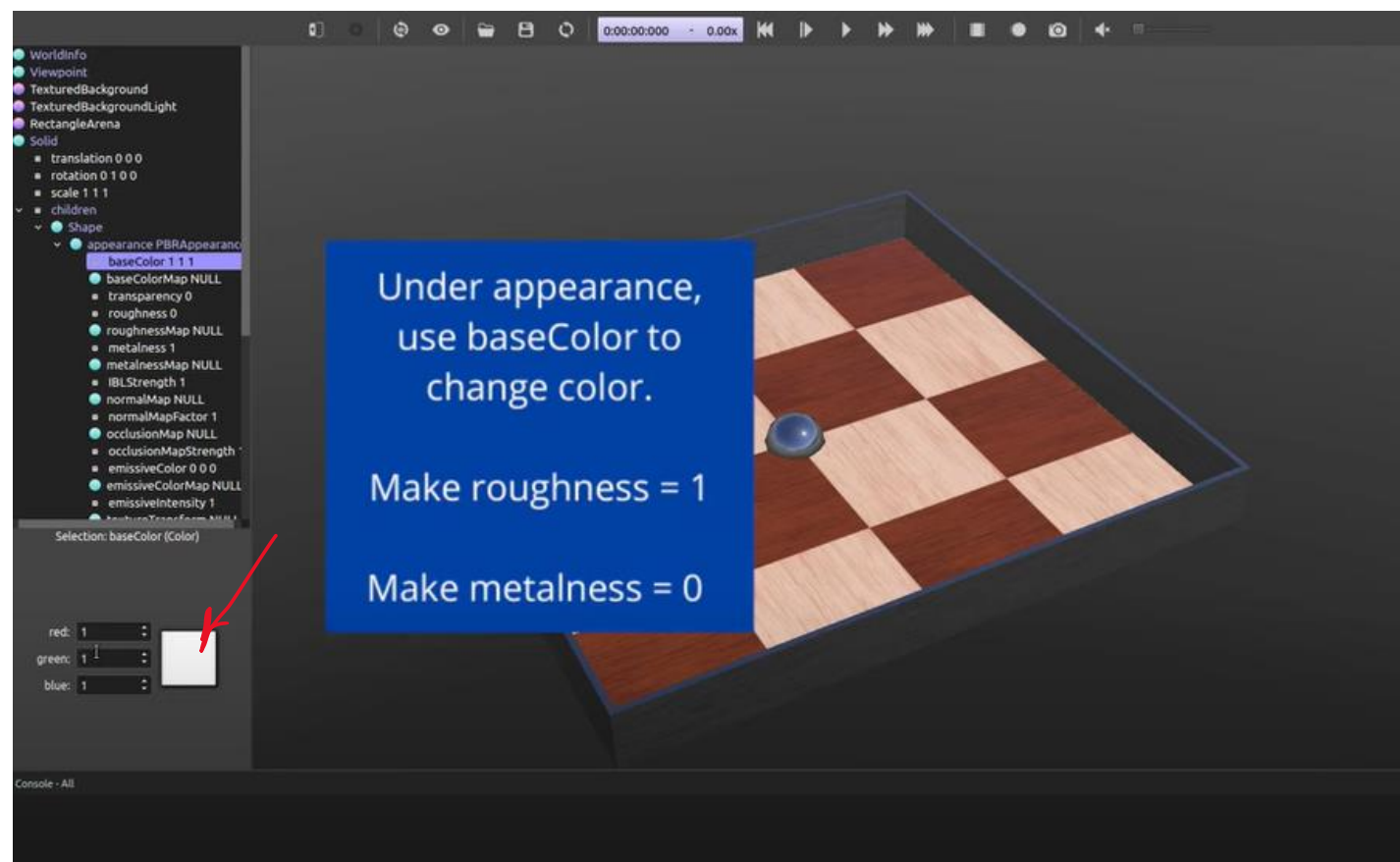
Chapter 03

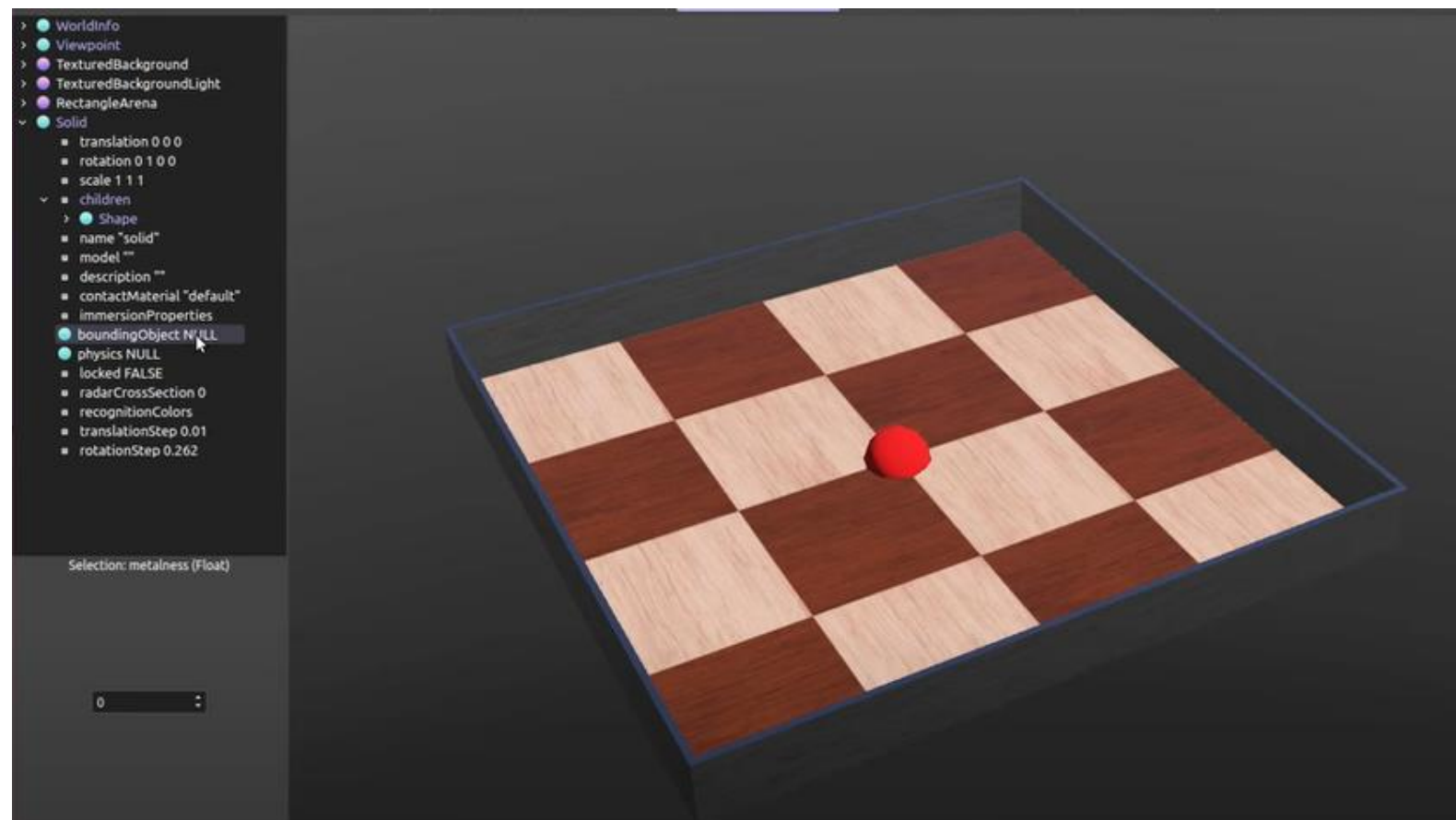
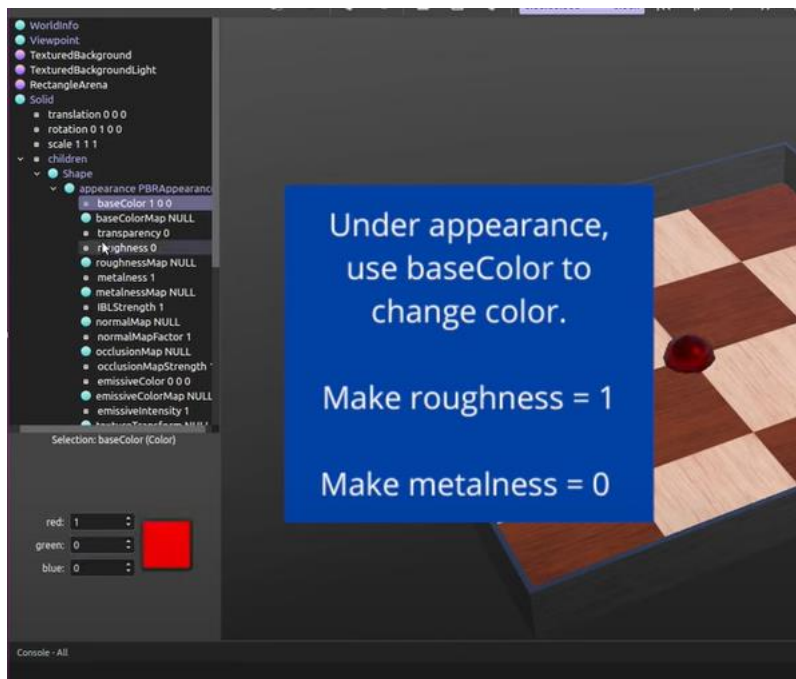
appearances



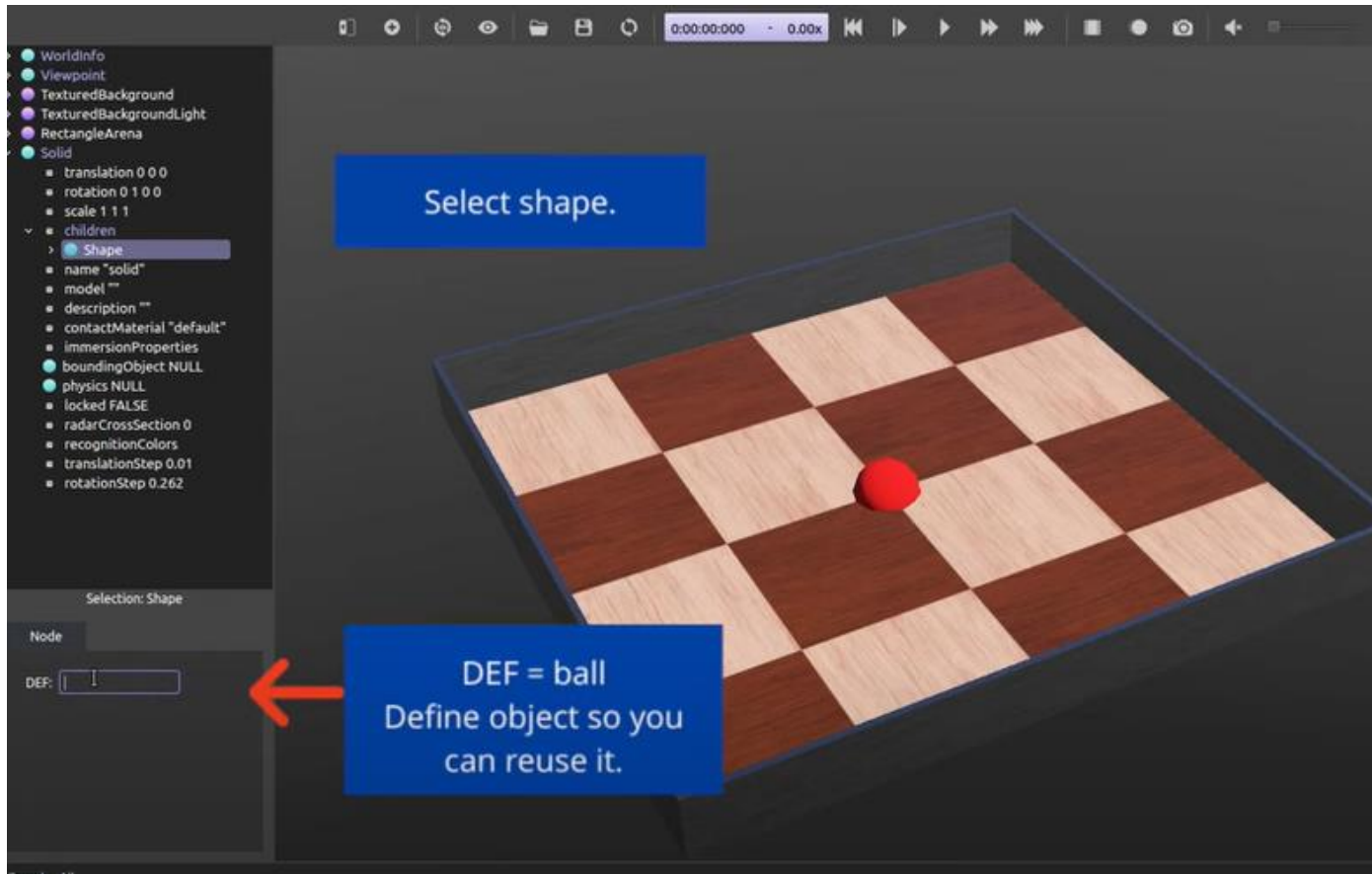


You have different options:



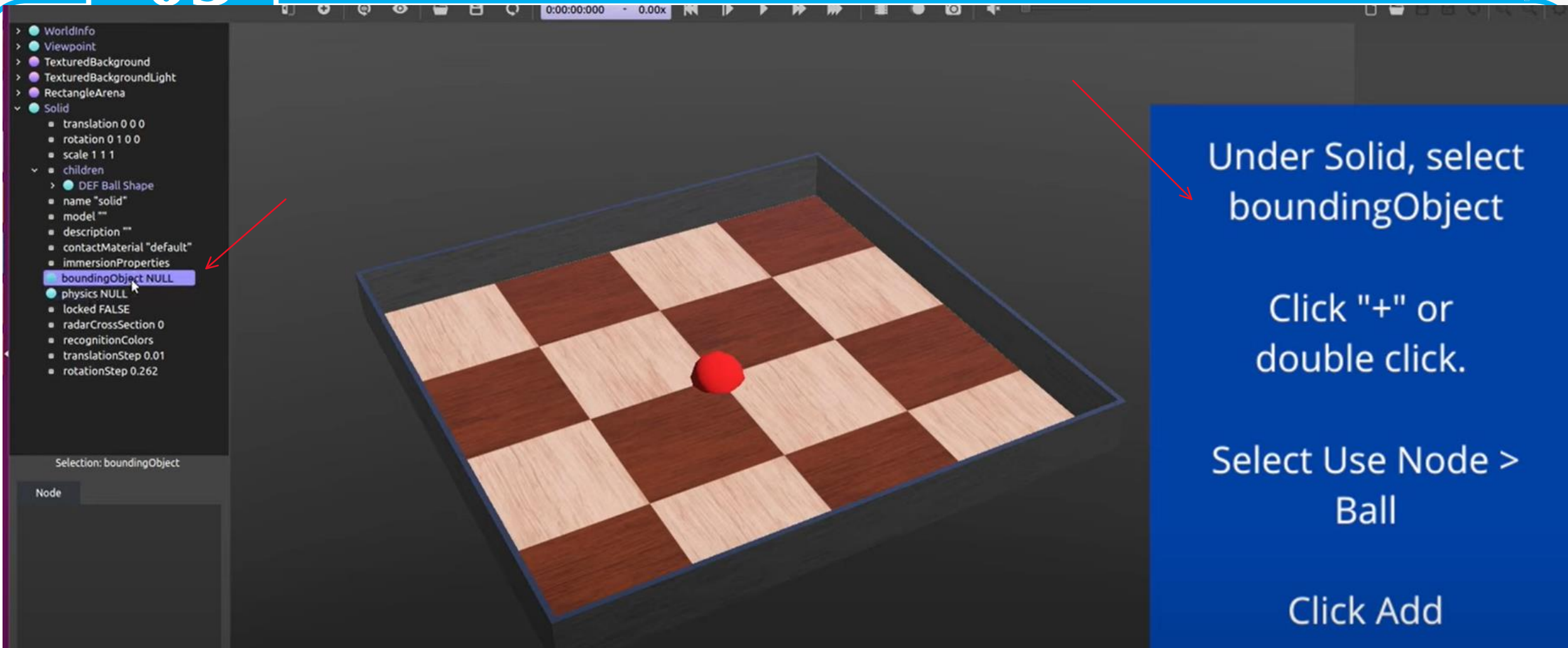


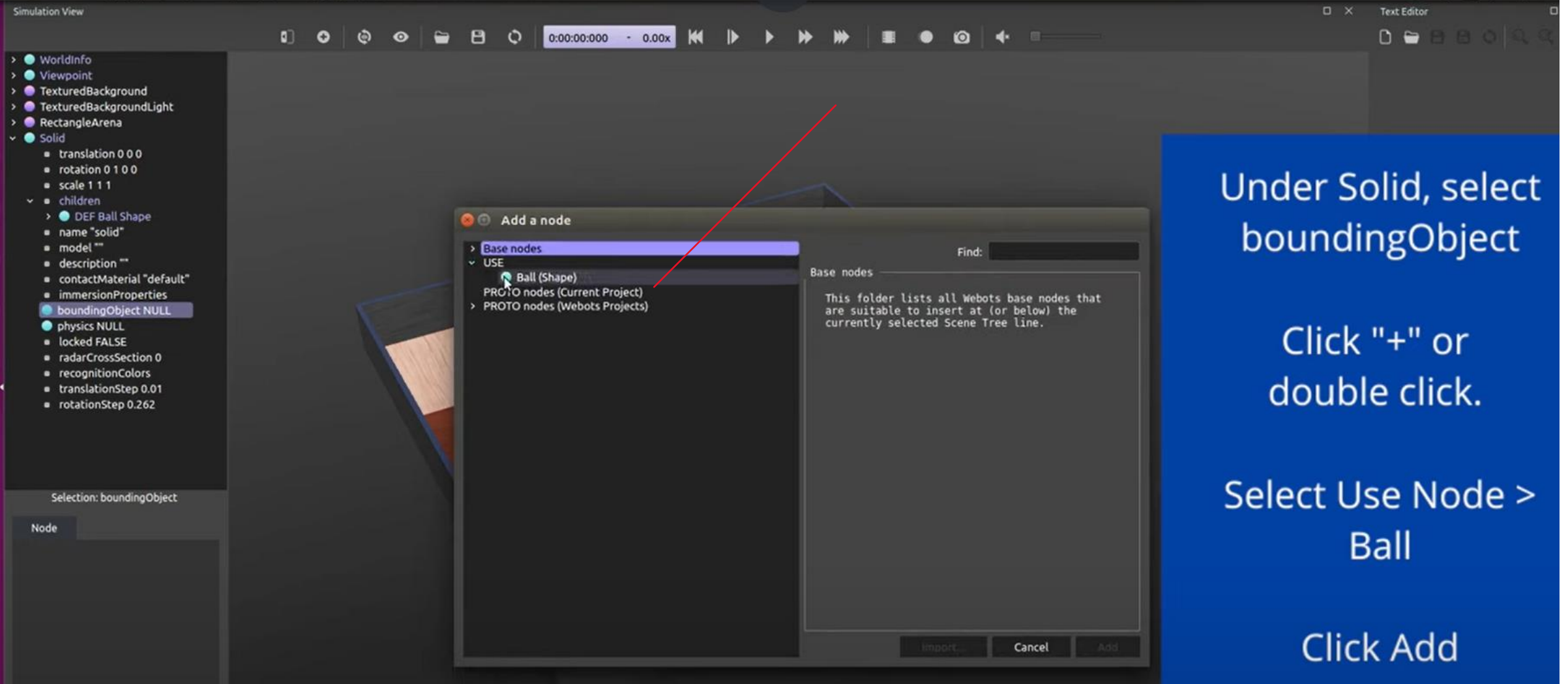
make it usable next we have to change the bounding object this is going to be the same thing as your shape so now instead of having to do it all over again you can reuse the same shape so click on shape and using def name it something that you can reuse so let's call it ball now on your bounding object when you double click you can click on use to use this shape that we just created which in this case is ball lastly let's add physics so in this case it's just a simple physics this helps with things like gravity and as you go into more advanced projects you can change this for different features



"DEF" refers to a "DEFinition" statement







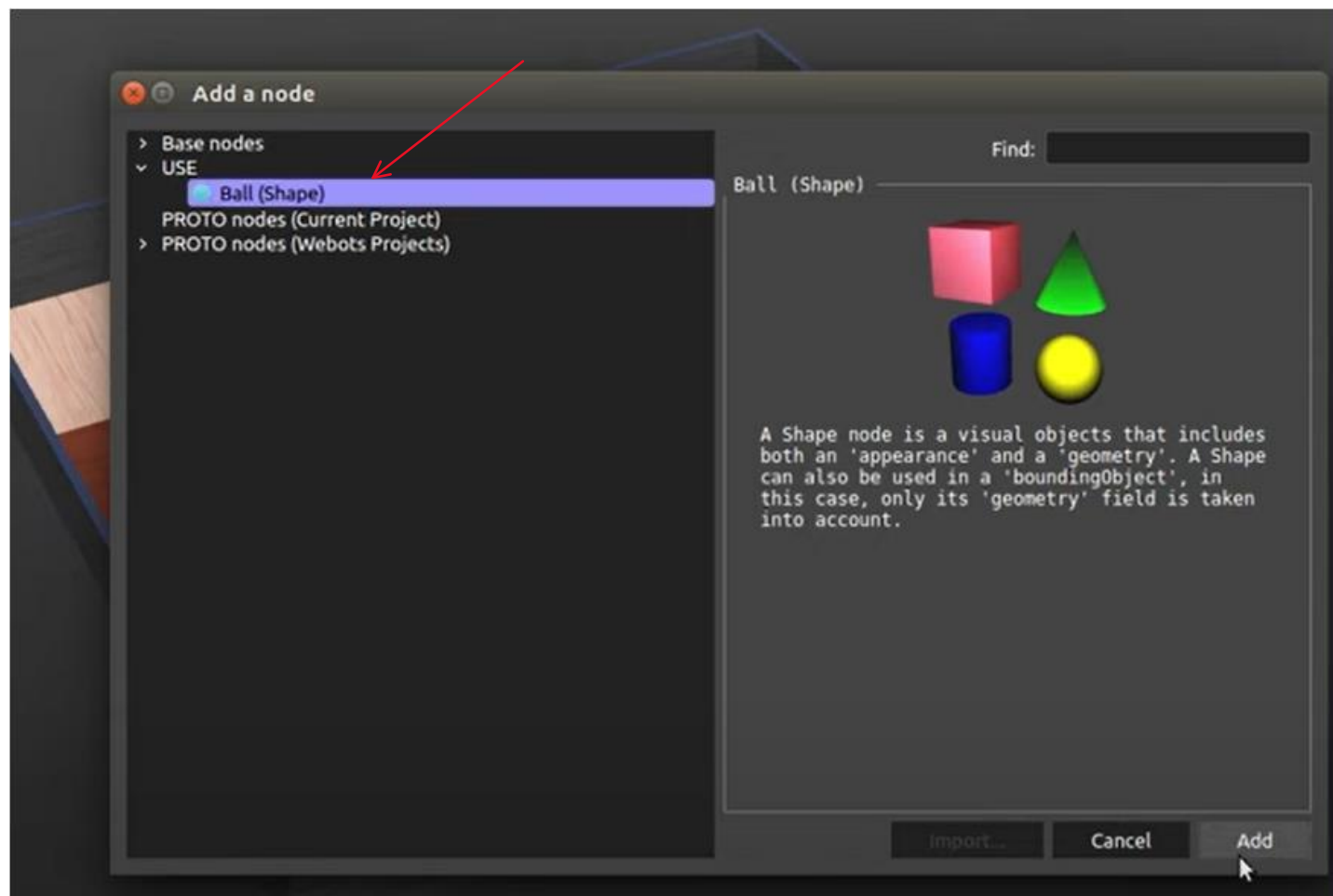
Under Solid, select
boundingObject

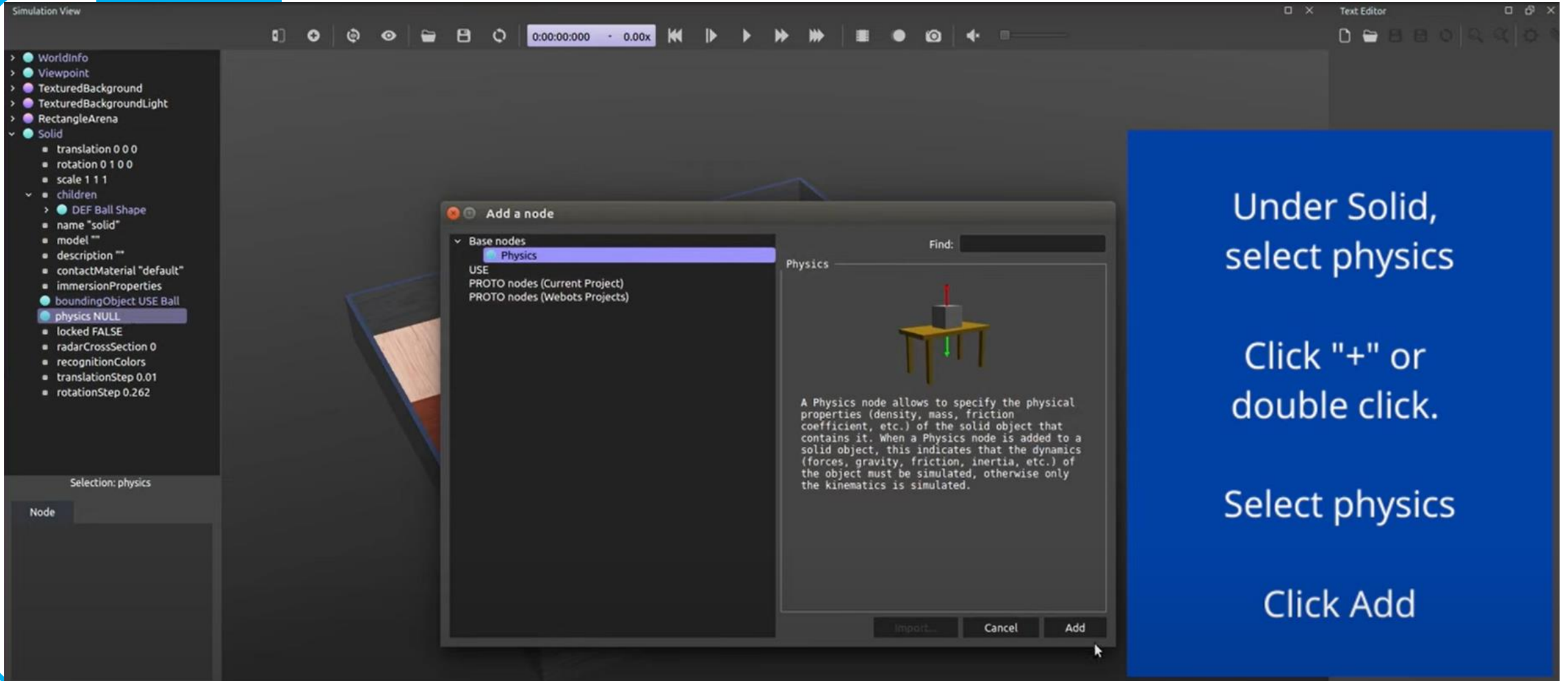
Click "+" or
double click.

Select Use Node >
Ball

Click Add

You can see the shape that you just create it





Simulation View

WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
RectangleArena
Solid
translation 0 0 0
rotation 0 1 0 0
scale 1 1 1
children
DEF Ball Shape
name "solid"
model ""
description ""
contactMaterial "default"
immersionProperties
boundingObject USE Ball
physics NULL
locked FALSE
radarCrossSection 0
recognitionColors
translationStep 0.01
rotationStep 0.262

Selection: physics

Node

Add a node

Base nodes
Physics
USE
PROTO nodes (Current Project)
PROTO nodes (Webots Projects)

Find:

Physics

A Physics node allows to specify the physical properties (density, mass, friction coefficient, etc.) of the solid object that contains it. When a Physics node is added to a solid object, this indicates that the dynamics (forces, gravity, friction, inertia, etc.) of the object must be simulated, otherwise only the kinematics is simulated.

Import... Cancel Add

Under Solid,
select physics

Click "+" or
double click.

Select physics

Click Add

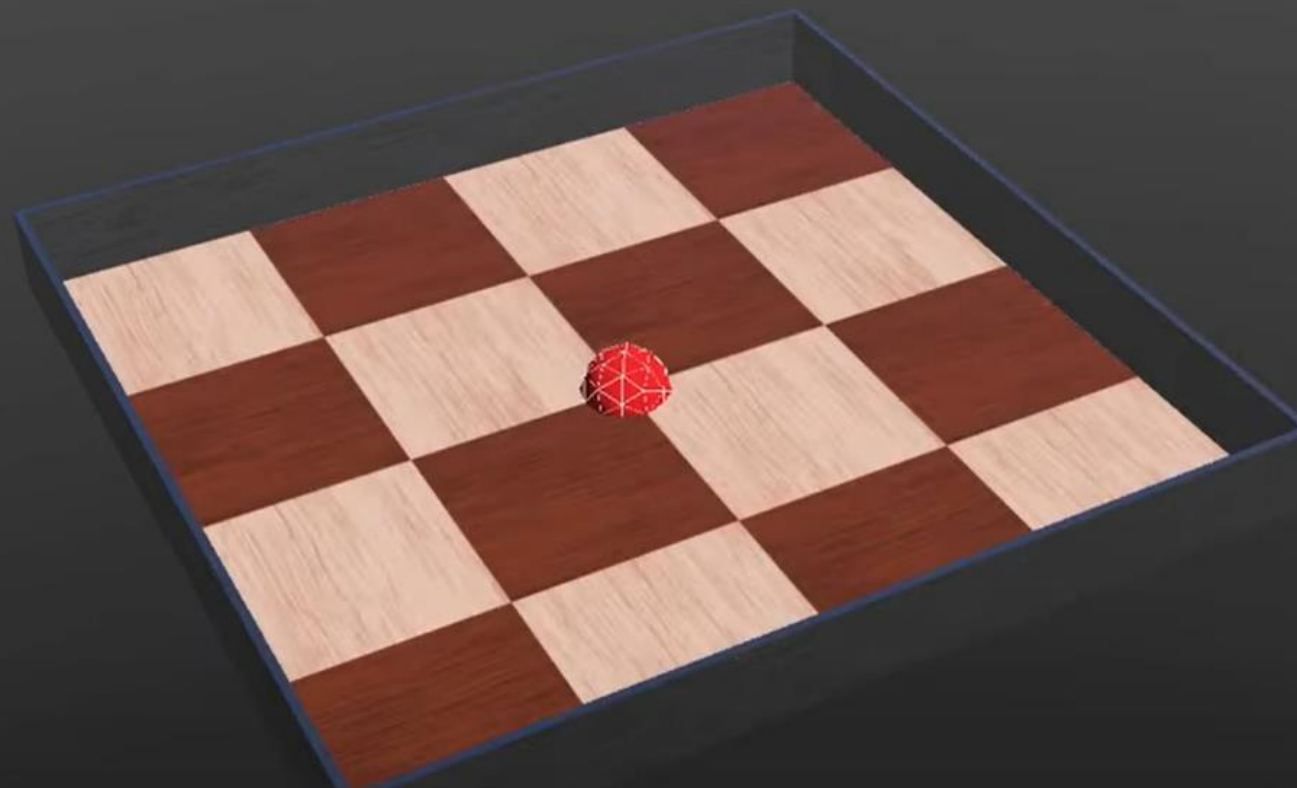
Now we will add a simple physics:

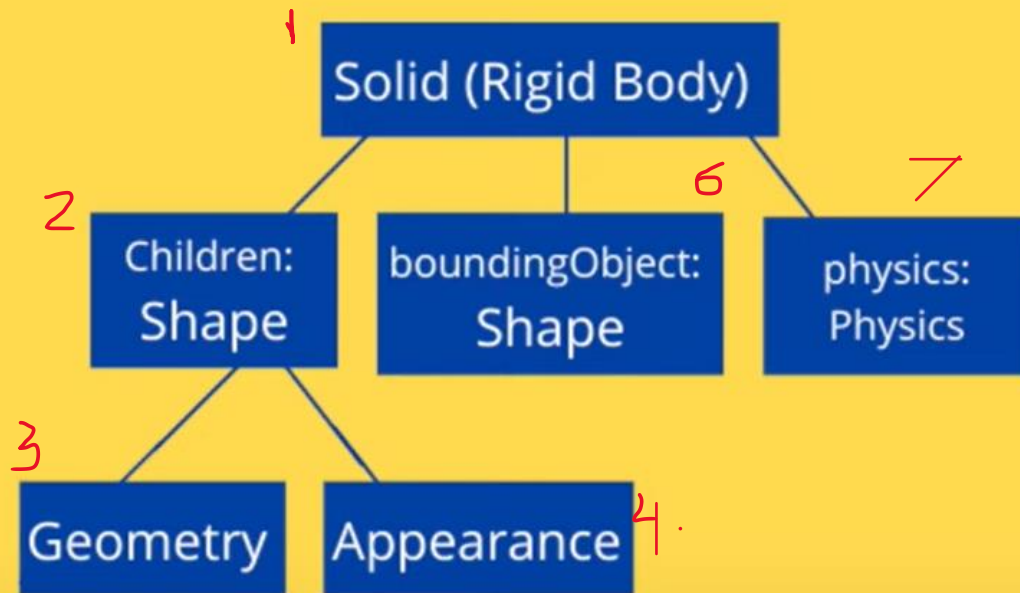
```
> WorldInfo
> Viewpoint
> TexturedBackground
> TexturedBackgroundLight
> RectangleArena
> Solid
  translation 0 0 0
  rotation 0 1 0 0
  scale 1 1 1
  children
    > DEF Ball Shape
      name "solid"
      model ""
      description ""
      contactMaterial "default"
      immersionProperties
      boundingObject USE Ball
    > physics Physics
      locked FALSE
      radarCrossSection 0
      recognitionColors
      translationStep 0.01
      rotationStep 0.262
```

Selection: physics

Node

DEF:





Creating an object

- (1) Start with a solid node.
- (2) Add a child node to give it shape.
- (3) Use geometry to choose the shape.
- (4) Add appearance and modify it.
- (5) Name shape using DEF for reuse.
- (6) Use DEF shape as bounding object
- (7) Add physics.

Creating an object in Webots involves a series of steps within its graphical interface, focusing on hierarchical node management and property adjustments. Here's a concise overview of the process:

1.Start with a Solid Node:

1. The initial step is to create a Solid node, which acts as a representation of a rigid body within the simulation environment. This is done by clicking the plus icon and navigating to "Base Nodes" > "Solid" to add a Solid node.

2.Add a Child Node (Shape):

1. Within the Solid node, you add a child node to define the visual and physical form of your object. This is achieved by adding a "Shape" node as a child of the Solid node.

3.Define the Geometry:

1. Under the Shape node, you specify the object's geometry. If you're creating a sphere, for instance, you select "Geometry" and then choose the "Sphere" option. This geometry can be customized further by modifying parameters such as the sphere's radius to alter its size.

4. Set the Appearance:

To define the object's visual characteristics, you click on "Appearance" and choose "PBRAppearance" as the base node. PBR (Physically Based Rendering) Appearance allows you to adjust visual properties like color, roughness, and metalness to achieve the desired look.

5. Define the Object for Reuse (DEF):

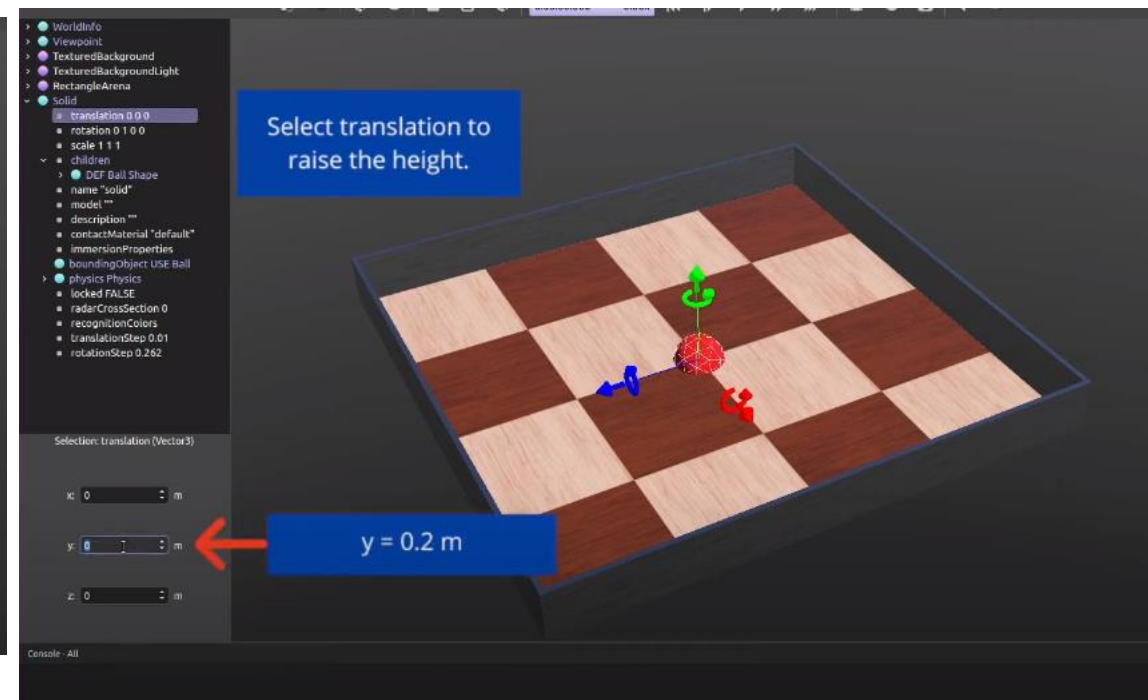
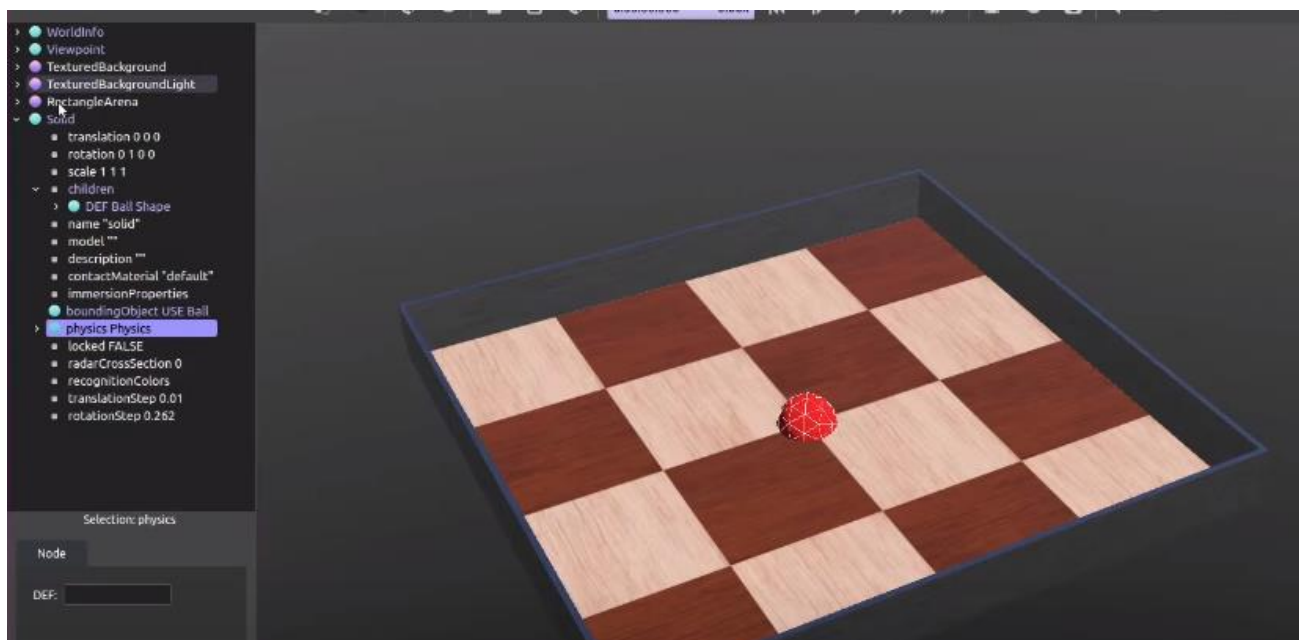
If you intend to reuse the shape within your simulation, you can define it as an object using the "DEF" keyword. This step involves giving a unique name to your shape, which allows you to reference and reuse it elsewhere in your simulation.

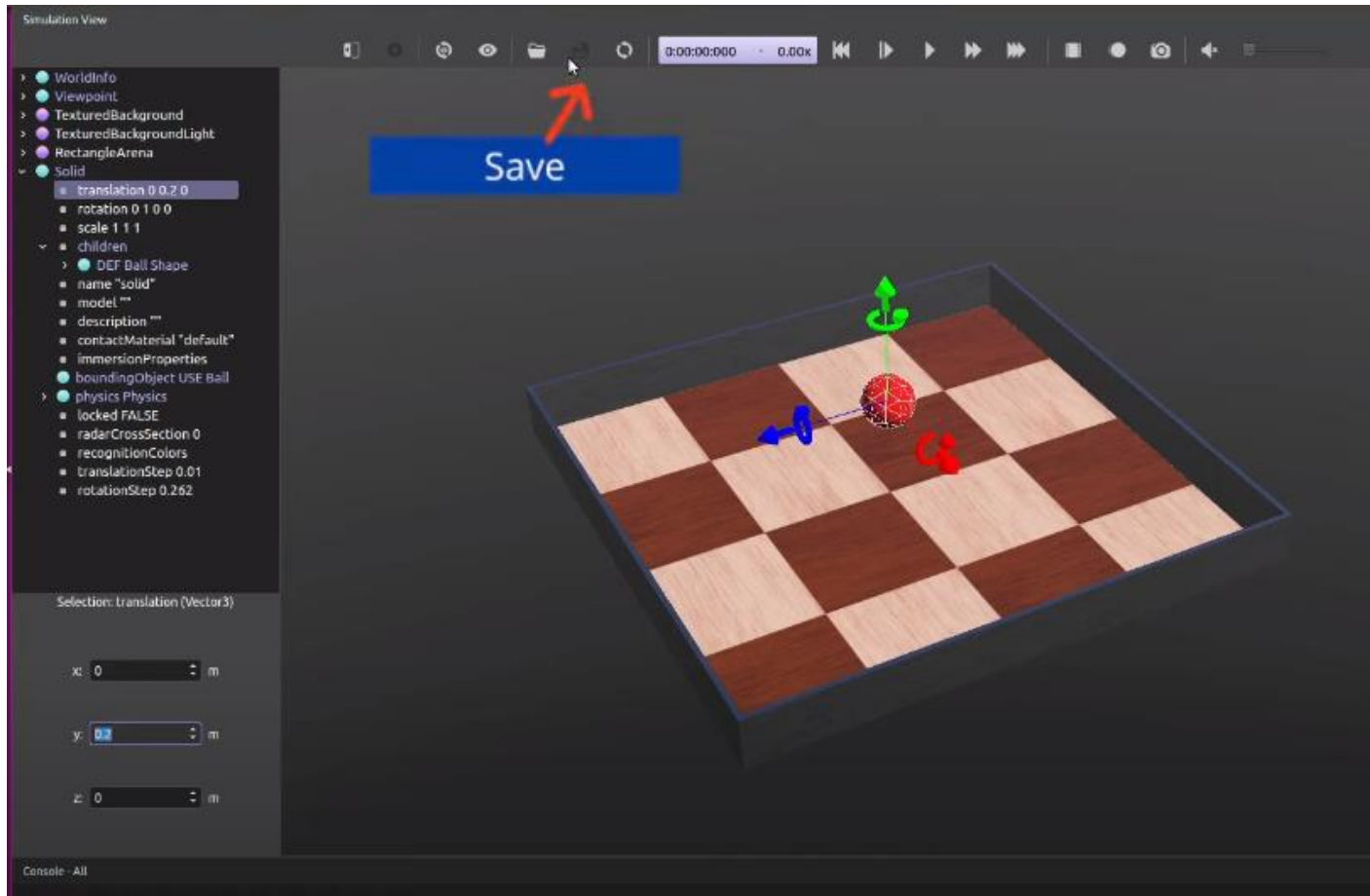
6. Specify the Bounding Object:

The bounding object is crucial for defining the collision boundaries of your object. You can set the Shape node you've already defined as the bounding object to ensure that collision detection works as expected within the simulation environment.

7. Add Physics:

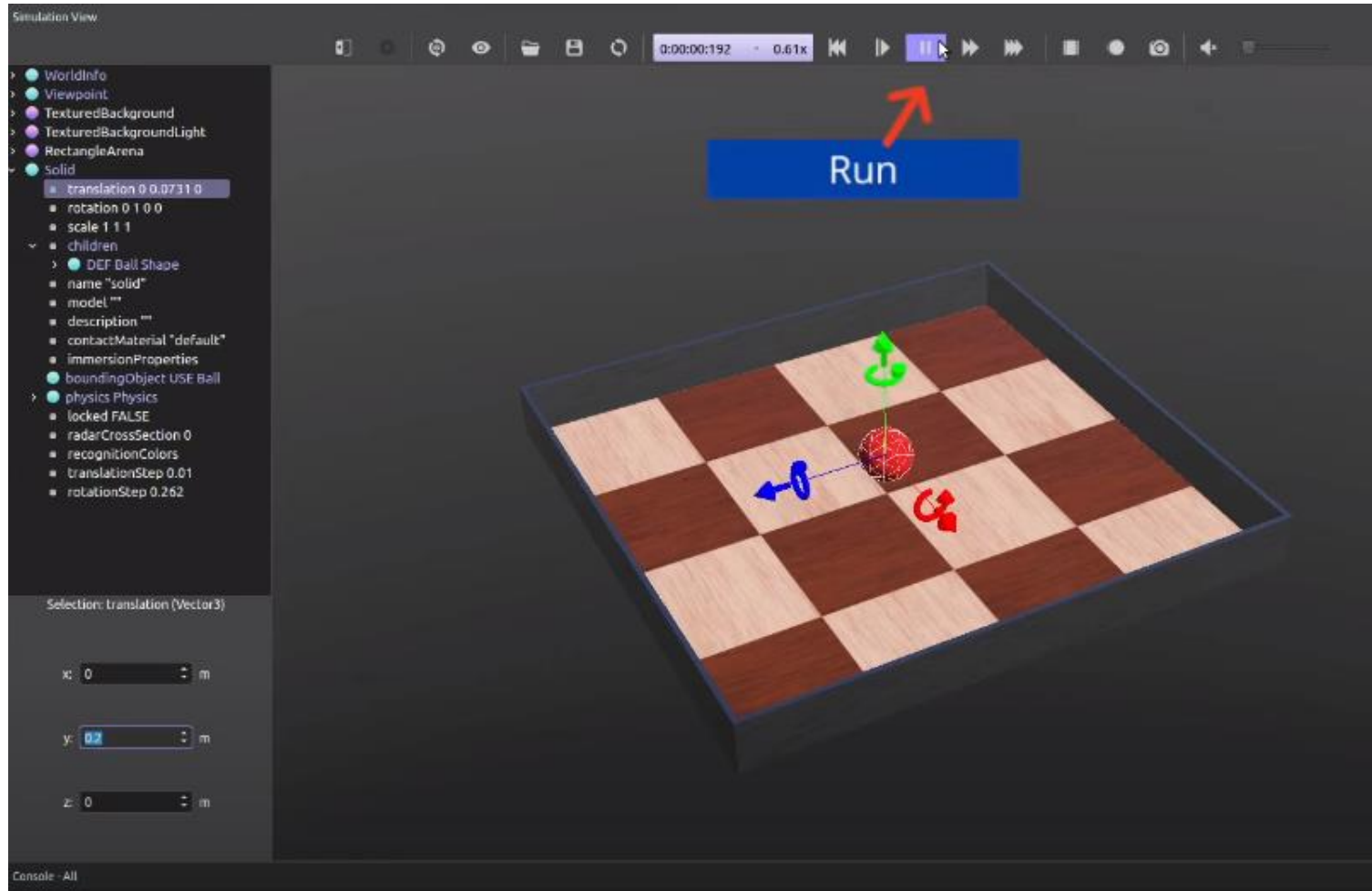
To make your object dynamic (capable of moving and interacting with other objects and the environment), you add physics properties to it. This distinguishes it from static elements in the simulation, like a rectangular arena, by enabling it to have physical behaviors such as bouncing, rolling, or colliding.



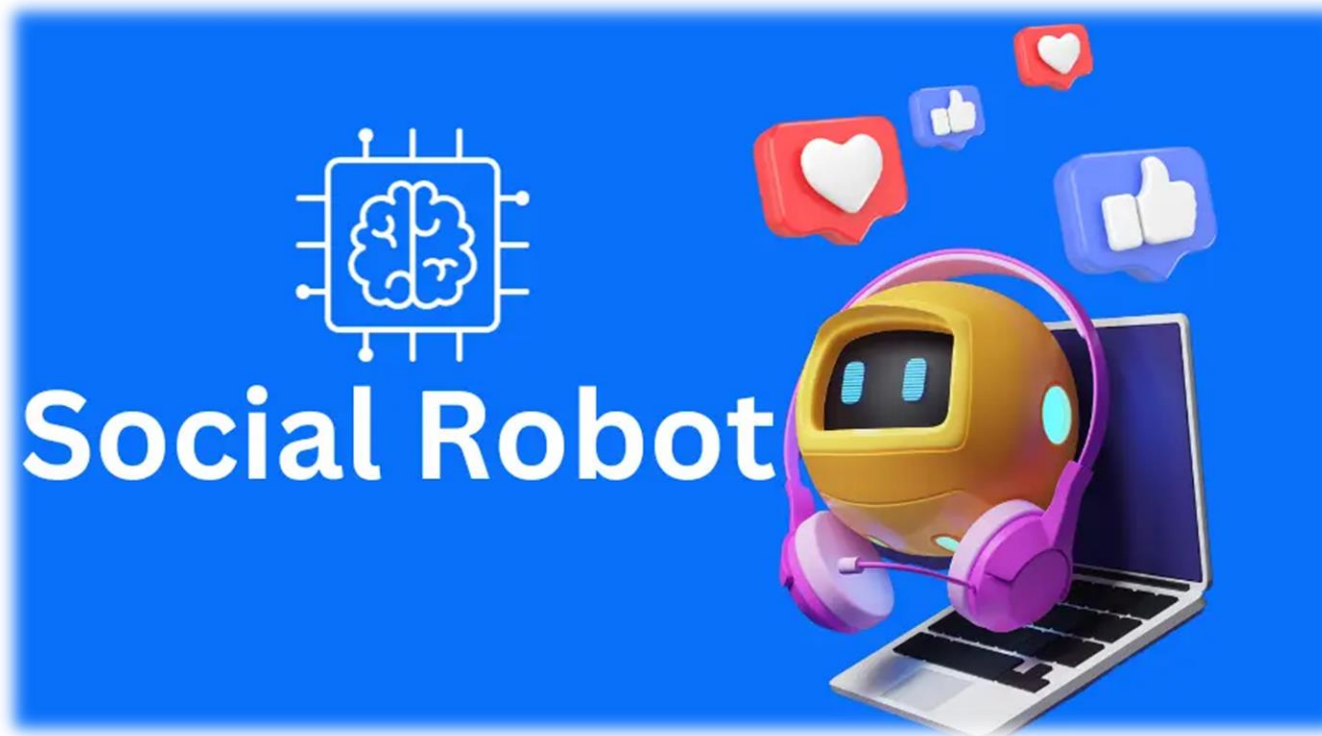


RUN the simulation:

The ball will drop on the floor, because the flood is static



What Are Social Robots?



Understanding Social Robots

Definition of social robots as machines designed to engage with humans in a social and interactive manner. Introduction to the concept of robots not just performing tasks but also relating to humans on a social level.



<https://www.aventine.org/robotics/history-of-robotics>



Overview of the sensors and AI technologies that enable social robots to perceive and understand human emotions, intentions, and social cues.

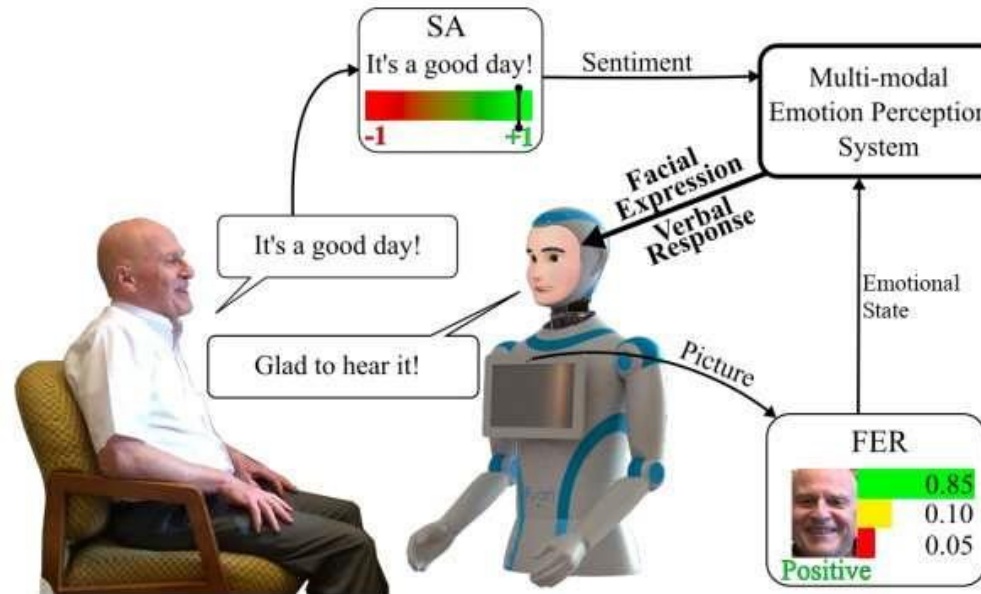


Fig. 1. Using a multimodal emotion perception system to interpret the input modalities and output appropriate responses in a multimodal emotion expression system (SA: Sentiment Analysis, FER: Facial Expression Recognition).

Exploring how social robots use AI to recognize and respond to human emotions, with examples like Jibo and Zenbo.



Zenbo.



Jibo - ROBOTS

How social robots assist in healthcare settings, such as providing companionship to the elderly or support to children with autism. Example: PARO therapeutic robot.



Paro is a robotic baby harp seal designed as a therapeutic tool for use in hospitals and nursing homes.



Overview of the use of social robots in educational settings, from language tutoring to STEM education. Example: The use of robots like Cozmo in teaching coding.

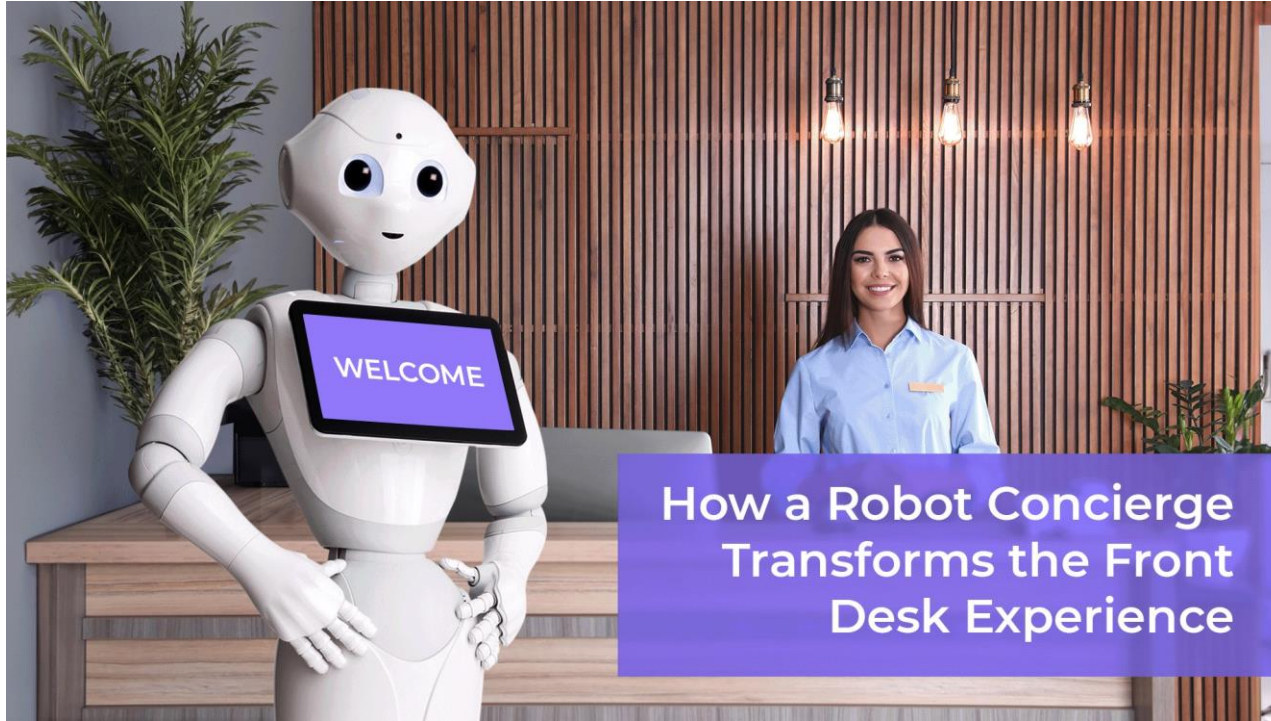
COZMO
+ Code Lab



anxi



social robots in customer service and hospitality, such as greeting guests or providing information. Example: Robots in hotels and retail.



How a Robot Concierge
Transforms the Front
Desk Experience



The technical, social, and ethical challenges in developing and integrating social robots into society.

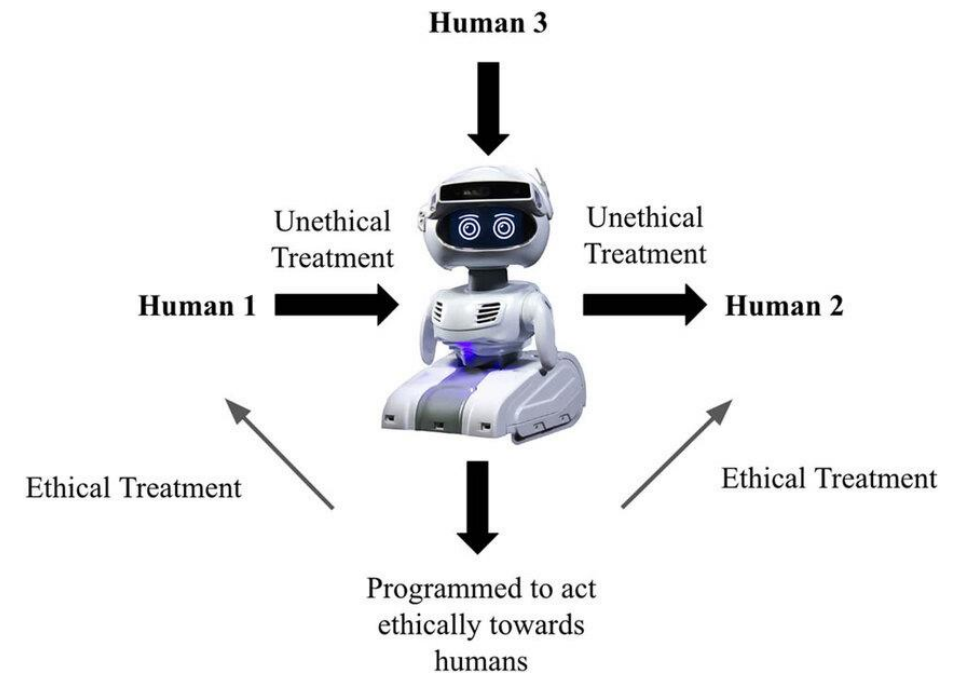


Examination of ethical issues, including privacy, autonomy, and the impact on employment. Questions of dependency and the role of robots in society.

Privacy Concerns:

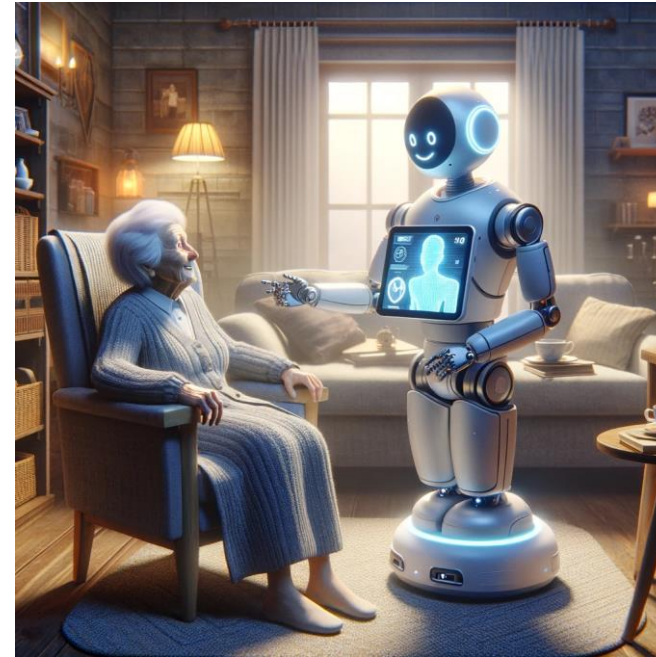
Autonomy and Human Agency:

Impact on Employment and Social Structures:



Privacy Concerns:

A home assistant robot that learns household routines and personal preferences could unintentionally expose sensitive information without robust data protection measures.



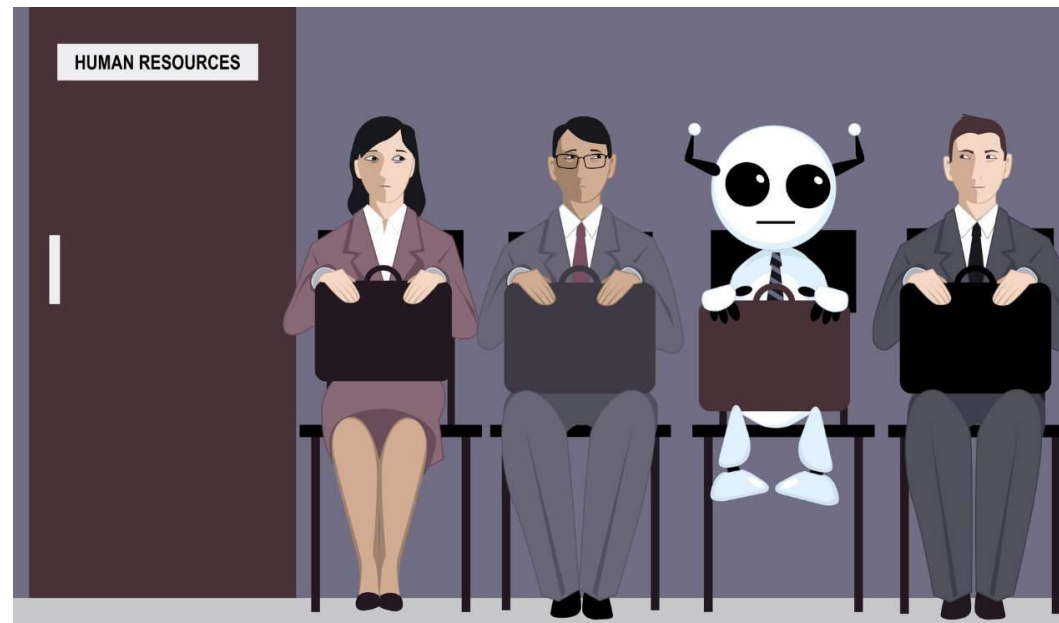
Autonomy and Human Agency:

Robots in elderly care settings making choices for residents about daily activities, thereby reducing their independence and decision-making capabilities.



Impact on Employment and Social Structures:

The use of robots for teaching or caregiving roles could alter employment opportunities in these sectors, necessitating a re-evaluation of human workers' roles and skills development for future job markets.





مجموعة جامعة
الجميع الذكية
للتعليم القابضة

MIDOCEAN
UNIVERSITY



Thank You

