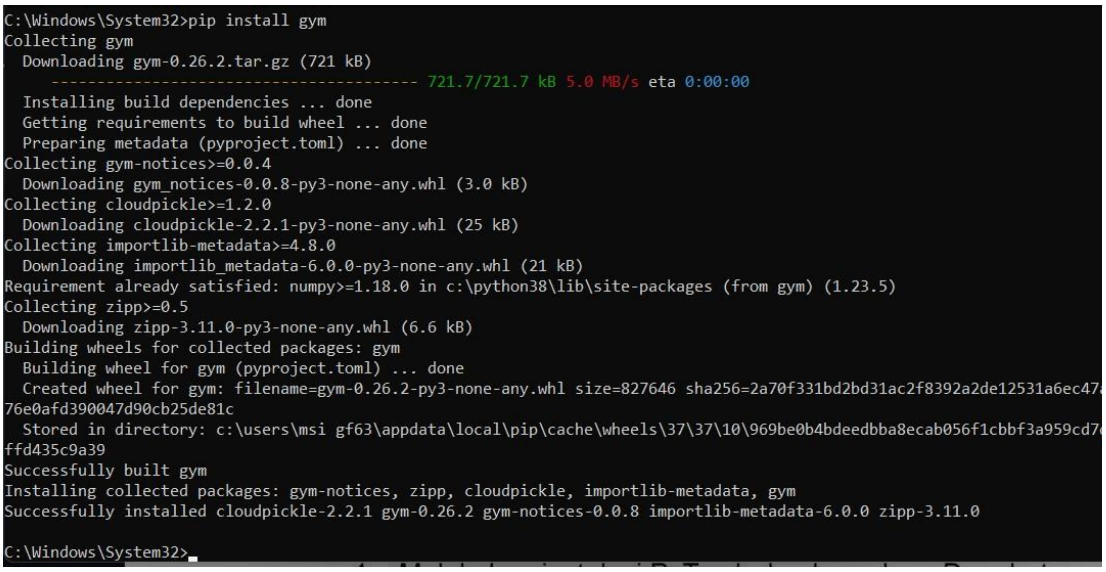
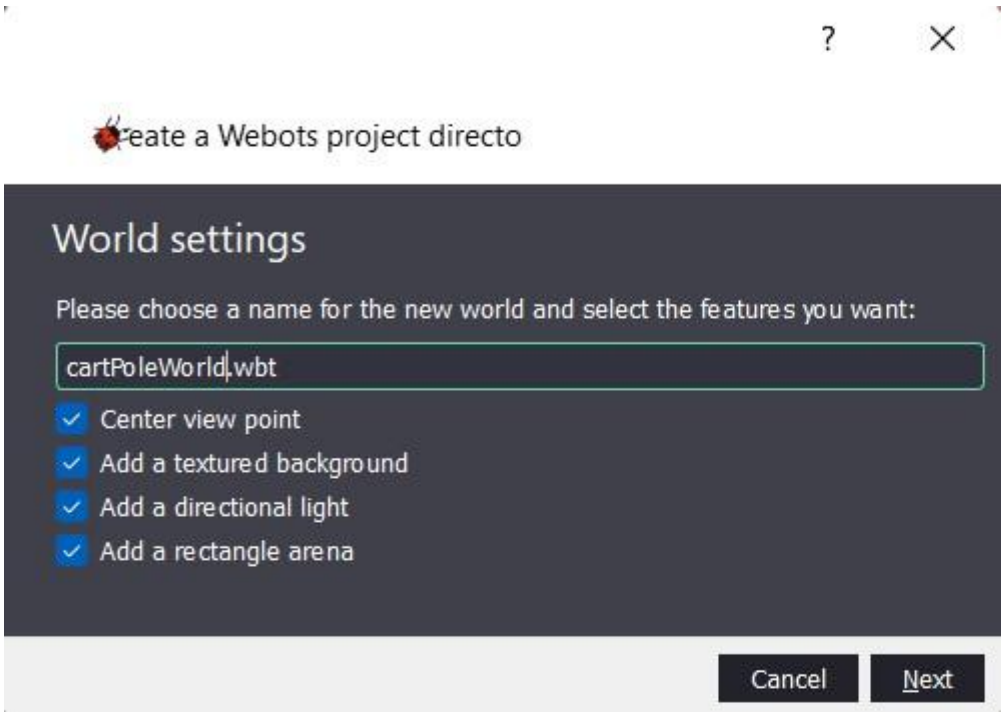
MUHAMMAD IQBAL DHARMAWAN | 1103194125 | UAS Robotic

1. Install pytorch



```
C:\Windows\System32>pip install gym
Collecting gym
  Downloading gym-0.26.2.tar.gz (721 kB)
     ---------------------------------------- 721.7/721.7 kB 5.0 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting gym-notices>=0.0.4
  Downloading gym_notices-0.0.8-py3-none-any.whl (3.0 kB)
Collecting cloudpickle>=1.2.0
  Downloading cloudpickle-2.2.1-py3-none-any.whl (25 kB)
Collecting importlib-metadata>=4.8.0
  Downloading importlib_metadata-6.0.0-py3-none-any.whl (21 kB)
Requirement already satisfied: numpy>=1.18.0 in c:\python38\lib\site-packages (from gym) (1.23.5)
Collecting zipp>=0.5
  Downloading zipp-3.11.0-py3-none-any.whl (6.6 kB)
Building wheels for collected packages: gym
  Building wheel for gym (pyproject.toml) ... done
  Created wheel for gym: filename=gym-0.26.2-py3-none-any.whl size=827646 sha256=2a70f331bd2bd31ac2f8392a2de12531a6ec47
76e0afd390047d90cb25de81c
  Stored in directory: c:\users\msi gf63\appdata\local\pip\cache\wheels\37\37\10\969be0b4bdeedbba8ecab056f1cbbf3a959cd7
ffd435c9a39
Successfully built gym
Installing collected packages: gym-notices, zipp, cloudpickle, importlib-metadata, gym
Successfully installed cloudpickle-2.2.1 gym-0.26.2 gym-notices-0.0.8 importlib-metadata-6.0.0 zipp-3.11.0

C:\Windows\System32>
```

2. Membuat project directory baru



eate a Webots project directo

World settings

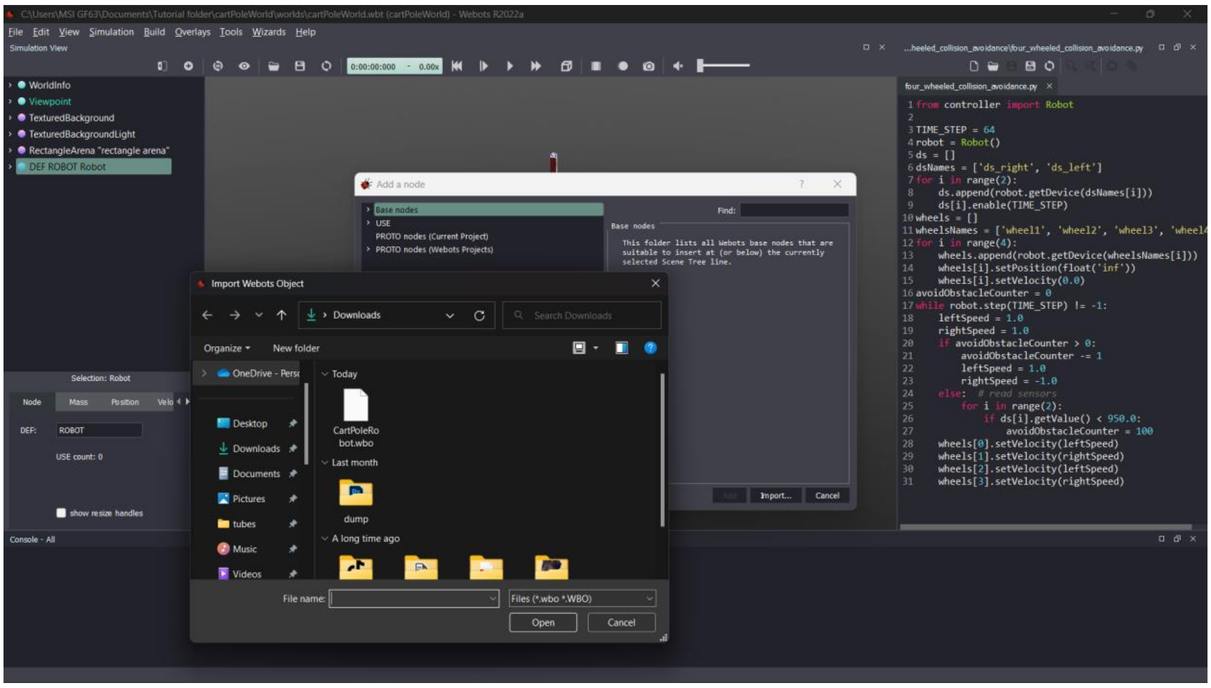
Please choose a name for the new world and select the features you want:

cartPoleWorld.wbt

- [x] Center view point
- [x] Add a textured background
- [x] Add a directional light
- [x] Add a rectangle arena

3. Download cartPole.wbo dari link yang ada pada halaman tutorial

4. Lalu Imort melalui add nodes

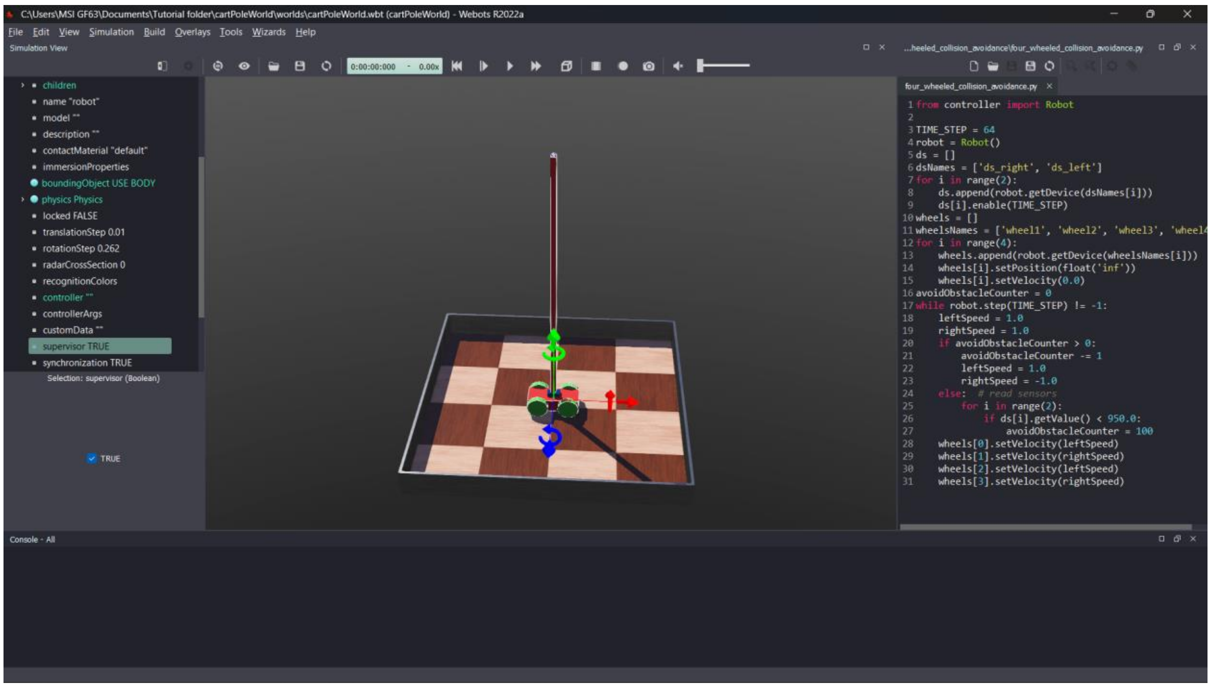


Base nodes

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

```python
from controller import Robot

TIME_STEP = 64
robot = Robot()
ds = []
dsNames = ['ds_right', 'ds_left']
for i in range(2):
    ds.append(robot.getDevice(dsNames[i]))
    ds[i].enable(TIME_STEP)
wheels = []
wheelsNames = ['wheel1', 'wheel2', 'wheel3', 'wheel4
for i in range(4):
    wheels.append(robot.getDevice(wheelsNames[i]))
    wheels[i].setPosition(float('inf'))
    wheels[i].setVelocity(0.0)
avoidObstacleCounter = 0
while robot.step(TIME_STEP) != -1:
    leftSpeed = 1.0
    rightSpeed = 1.0
    if avoidObstacleCounter > 0:
        avoidObstacleCounter -= 1
        leftSpeed = 1.0
        rightSpeed = -1.0
    else:  # read sensors
        for i in range(2):
            if ds[i].getValue() < 950.0:
                avoidObstacleCounter = 100
    wheels[0].setVelocity(leftSpeed)
    wheels[1].setVelocity(rightSpeed)
    wheels[2].setVelocity(leftSpeed)
    wheels[3].setVelocity(rightSpeed)
```

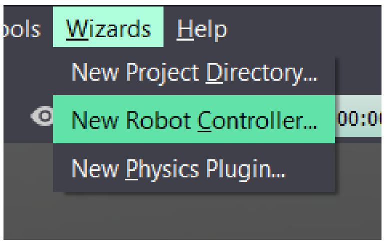
5. Setelah robot berhasil di import, maka ubah “supervisor” menjadi TRUE


C:\Users\MSI GF63\Documents\Tutorial folder\cartPoleWorld\worlds\cartPoleWorld.wbt (cartPoleWorld) - Webots R2022a
File Edit View Simulation Build Overlays Tools Wizards Help
Console - All
four_wheeled_collision_avoidance.py

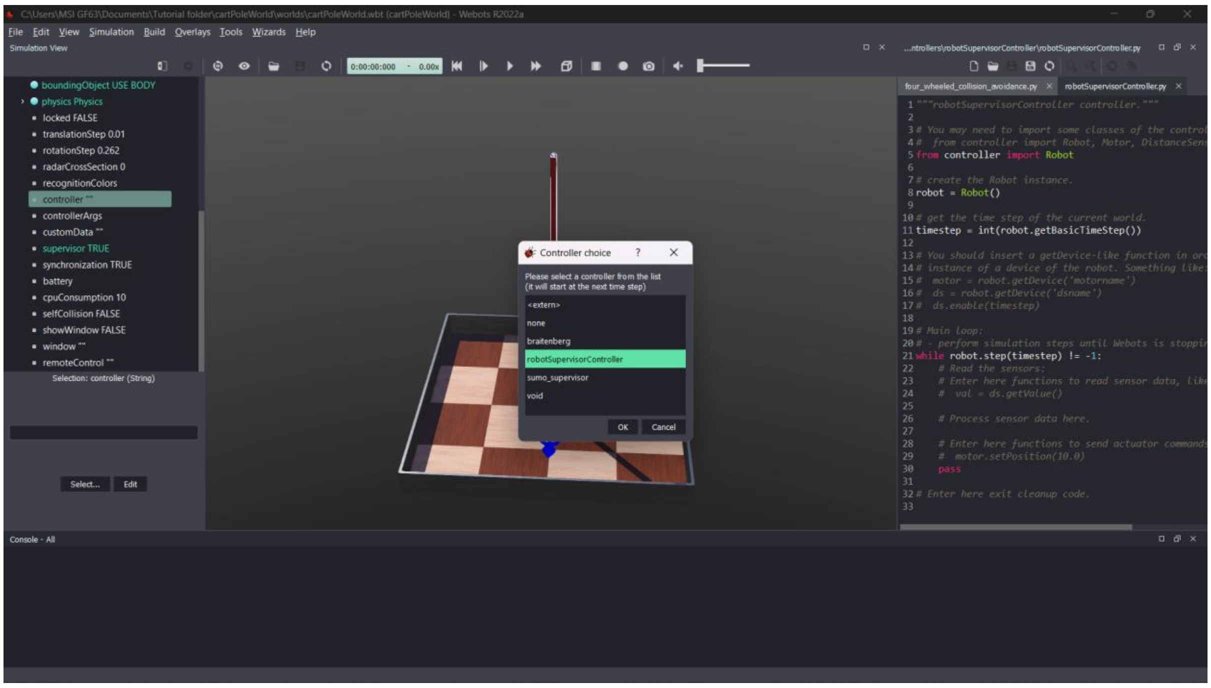
children
name "robot"
model ""
description ""
contactMaterial "default"
immersionProperties
boundingObject USE BODY
physics Physics
locked FALSE
translationStep 0.01
rotationStep 0.262
radarCrossSection 0
recognitionColors
controller ""
controllerArgs
customData ""
supervisor TRUE
synchronization TRUE
Selection: supervisor (Boolean)
✓ TRUE

```python
from controller import Robot

TIME_STEP = 64
robot = Robot()
ds = []
dsNames = ['ds_right', 'ds_left']
for i in range(2):
    ds.append(robot.getDevice(dsNames[i]))
    ds[i].enable(TIME_STEP)
wheels = []
wheelsNames = ['wheel1', 'wheel2', 'wheel3', 'wheel4
for i in range(4):
    wheels.append(robot.getDevice(wheelsNames[i]))
    wheels[i].setPosition(float('inf'))
    wheels[i].setVelocity(0.0)
avoidObstacleCounter = 0
while robot.step(TIME_STEP) != -1:
    leftSpeed = 1.0
    rightSpeed = 1.0
    if avoidObstacleCounter > 0:
        avoidObstacleCounter -= 1
        leftSpeed = 1.0
        rightSpeed = -1.0
    else:  # read sensors
        for i in range(2):
            if ds[i].getValue() < 950.0:
                avoidObstacleCounter = 100
    wheels[0].setVelocity(leftSpeed)
    wheels[1].setVelocity(rightSpeed)
    wheels[2].setVelocity(leftSpeed)
    wheels[3].setVelocity(rightSpeed)
```

6. Tambahkan controller baru



7. Lalu select controller yang sudah dibuat


C:\Users\MSI GF63\Documents\Tutorial folder\cartPoleWorld\worlds\cartPoleWorld.wbt (cartPoleWorld) - Webots R2022a

```
boundingObject USE BODY
physics Physics
locked FALSE
translationStep 0.01
rotationStep 0.262
radarCrossSection 0
recognitionColors
controller ""
controllerArgs
customData ""
supervisor TRUE
synchronization TRUE
battery
cpuConsumption 10
selfCollision FALSE
showWindow FALSE
window ""
remoteControl ""
Selection: controller (String)
```

Controller choice

Please select a controller from the list
(it will start at the next time step)

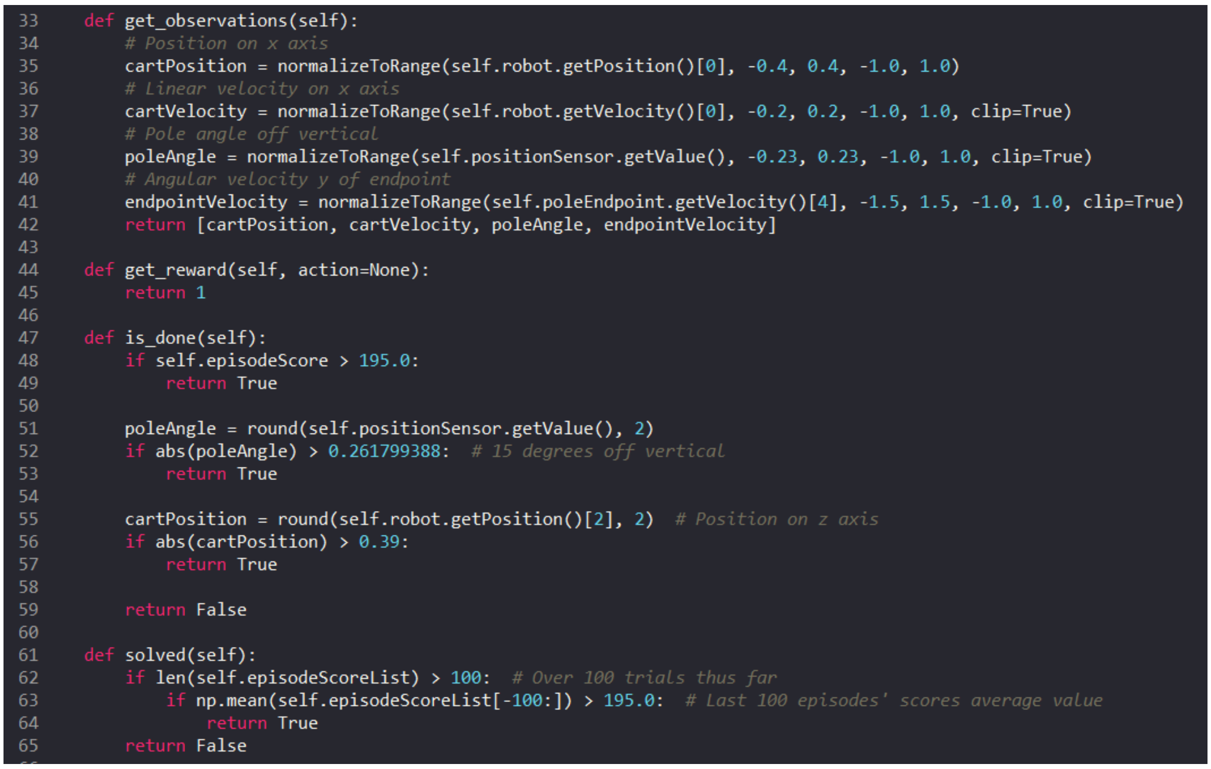
```
<extern>
none
braitenberg
robotSupervisorController
sumo_supervisor
void
```

...ntrollers\robotSupervisorController\robotSupervisorController.py

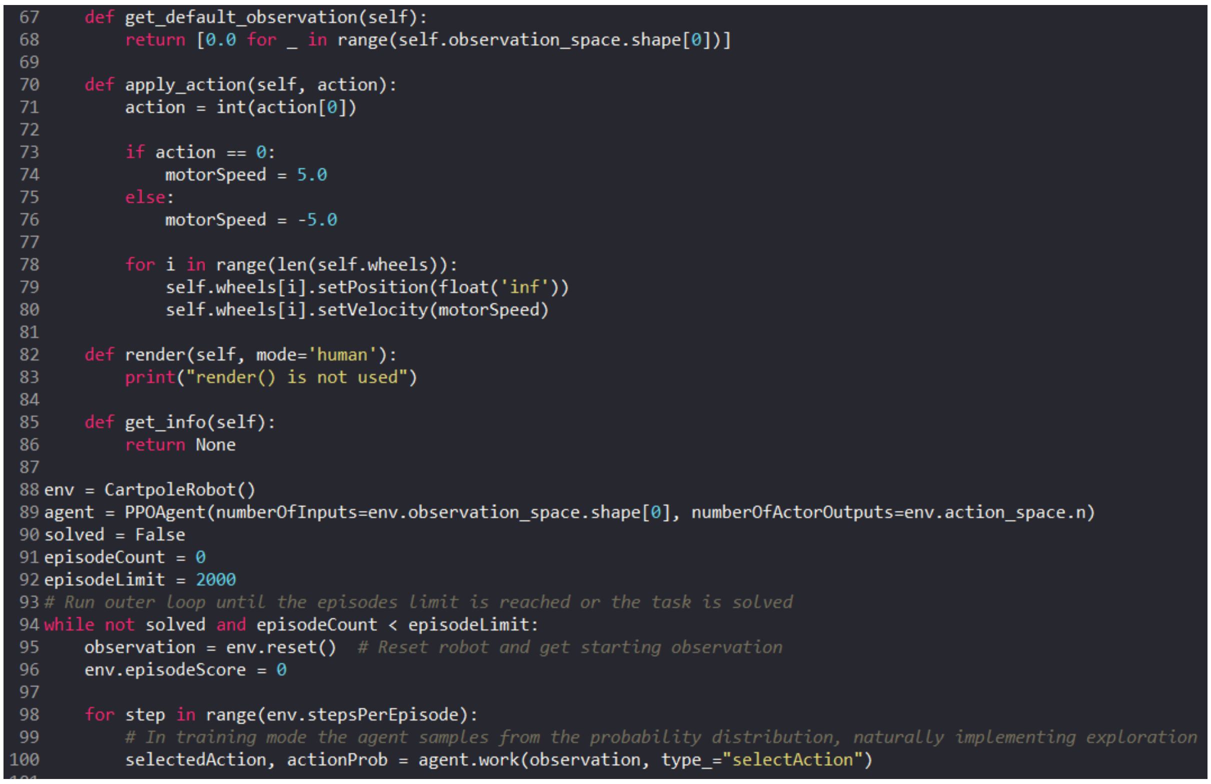
```python
"""robotSupervisorController controller."""

# You may need to import some classes of the control
#  from controller import Robot, Motor, DistanceSens
from controller import Robot

# create the Robot instance.
robot = Robot()

# get the time step of the current world.
timestep = int(robot.getBasicTimeStep())

# You should insert a getDevice-like function in ord
# instance of a device of the robot. Something like:
#  motor = robot.getDevice('motorname')
#  ds = robot.getDevice('dsname')
#  ds.enable(timestep)

# Main loop:
# - perform simulation steps until Webots is stoppi
while robot.step(timestep) != -1:
    # Read the sensors:
    # Enter here functions to read sensor data, like
    #  val = ds.getValue()

    # Process sensor data here.

    # Enter here functions to send actuator commands
    #  motor.setPosition(10.0)
    pass

# Enter here exit cleanup code.

```
File Edit View Simulation Build Overlays Tools Wizards Help
Simulation View
0:00:00:000 - 0.00x
four_wheeled_collision_avoidance.py
robotSupervisorController.py
Select...
Edit
OK
Cancel
Console - All

8. Lalu download PPO agent dan utilities script melalui web tutorial lalu menyimpannya di dalam folder controller/robotSupervisorController/.
9. Mulai menyusun codingan

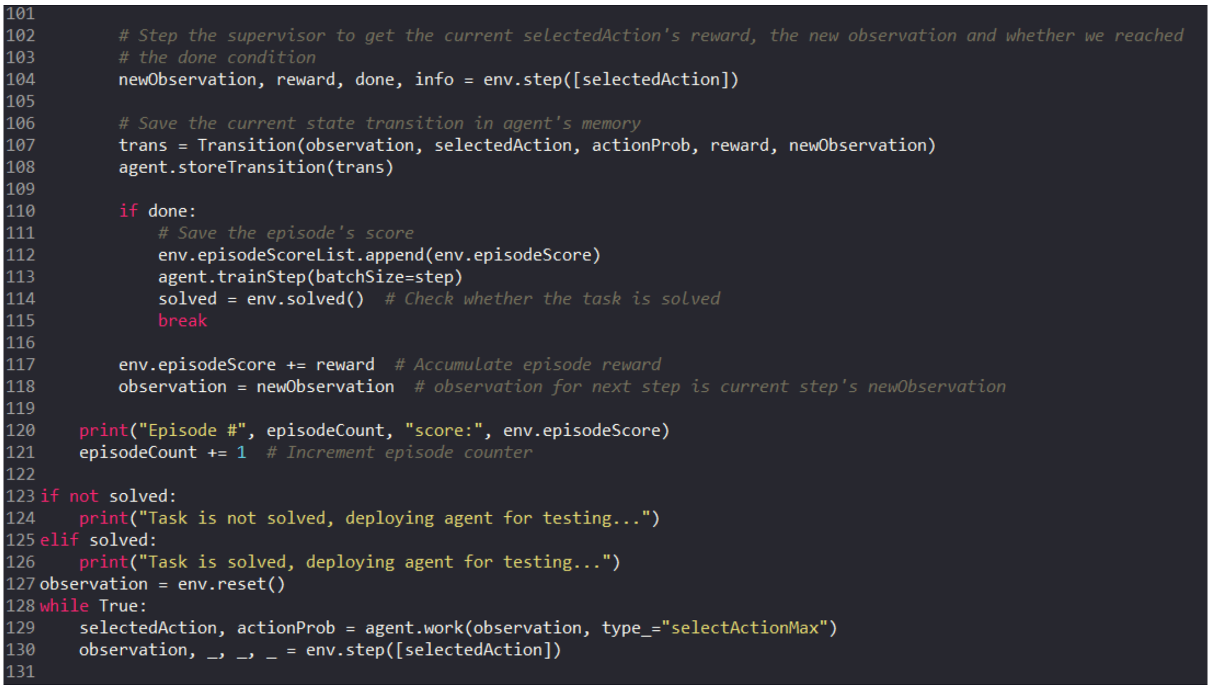
```python
from deepbots.supervisor.controllers.robot_supervisor import RobotSupervisor
from utilities import normalizeToRange, plotData
from PPO_agent import PPOAgent, Transition

from gym.spaces import Box, Discrete
import numpy as np


class CartpoleRobot(RobotSupervisor):
    def __init__(self):
        super().__init__()
        self.observation_space = Box(low=np.array([-0.4, -np.inf, -1.3, -np.inf]),
                                     high=np.array([0.4, np.inf, 1.3, np.inf]),
                                     dtype=np.float64)
        self.action_space = Discrete(2)

        self.robot = self.getSelf()  # Grab the robot reference from the supervisor to access various robot methods
        self.positionSensor = self.getDevice("polePosSensor")
        self.positionSensor.enable(self.timestep)

        self.poleEndpoint = self.getFromDef("POLE_ENDPOINT")
        self.wheels = []
        for wheelName in ['wheel1', 'wheel2', 'wheel3', 'wheel4']:
            wheel = self.getDevice(wheelName)  # Get the wheel handle
            wheel.setPosition(float('inf'))  # Set starting position
            wheel.setVelocity(0.0)  # Zero out starting velocity
            self.wheels.append(wheel)

        self.stepsPerEpisode = 200  # Max number of steps per episode
        self.episodeScore = 0  # Score accumulated during an episode
        self.episodeScoreList = []  # A list to save all the episode scores, used to check if task is solved
```



```python
    def get_observations(self):
        # Position on x axis
        cartPosition = normalizeToRange(self.robot.getPosition()[0], -0.4, 0.4, -1.0, 1.0)
        # Linear velocity on x axis
        cartVelocity = normalizeToRange(self.robot.getVelocity()[0], -0.2, 0.2, -1.0, 1.0, clip=True)
        # Pole angle off vertical
        poleAngle = normalizeToRange(self.positionSensor.getValue(), -0.23, 0.23, -1.0, 1.0, clip=True)
        # Angular velocity y of endpoint
        endpointVelocity = normalizeToRange(self.poleEndpoint.getVelocity()[4], -1.5, 1.5, -1.0, 1.0, clip=True)
        return [cartPosition, cartVelocity, poleAngle, endpointVelocity]

    def get_reward(self, action=None):
        return 1

    def is_done(self):
        if self.episodeScore > 195.0:
            return True

        poleAngle = round(self.positionSensor.getValue(), 2)
        if abs(poleAngle) > 0.261799388:  # 15 degrees off vertical
            return True

        cartPosition = round(self.robot.getPosition()[2], 2)  # Position on z axis
        if abs(cartPosition) > 0.39:
            return True

        return False

    def solved(self):
        if len(self.episodeScoreList) > 100:  # Over 100 trials thus far
            if np.mean(self.episodeScoreList[-100:]) > 195.0:  # Last 100 episodes' scores average value
                return True
        return False
```

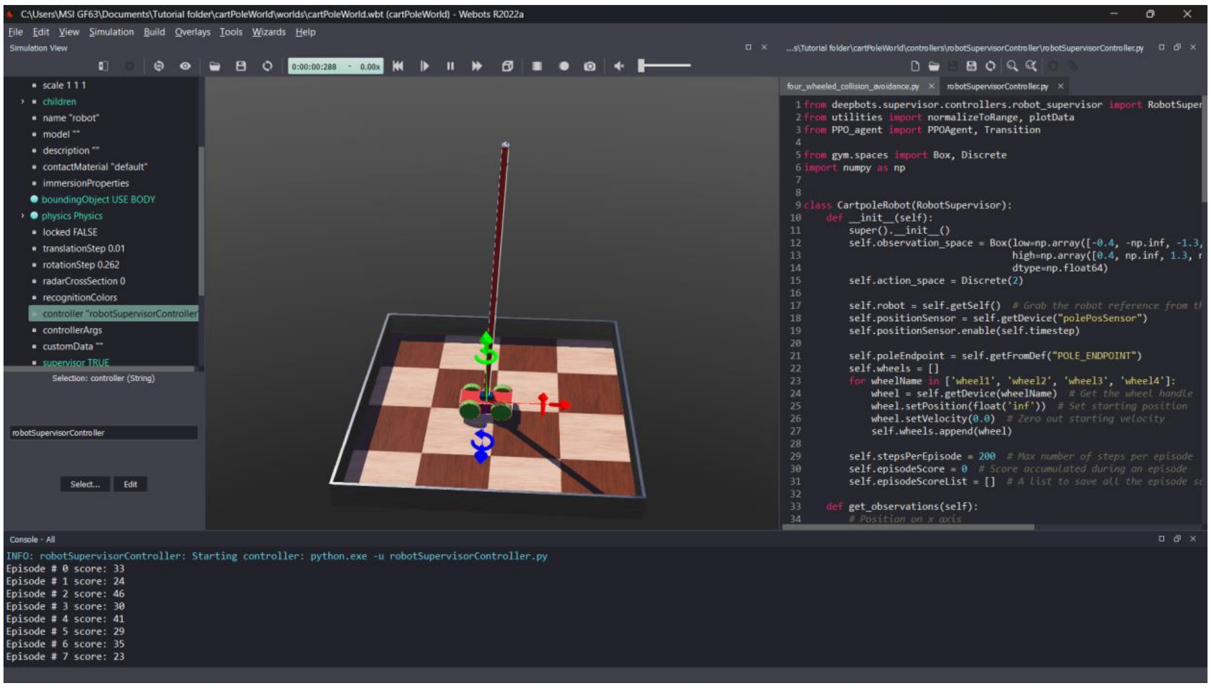


```python
    def get_default_observation(self):
        return [0.0 for _ in range(self.observation_space.shape[0])]

    def apply_action(self, action):
        action = int(action[0])

        if action == 0:
            motorSpeed = 5.0
        else:
            motorSpeed = -5.0

        for i in range(len(self.wheels)):
            self.wheels[i].setPosition(float('inf'))
            self.wheels[i].setVelocity(motorSpeed)

    def render(self, mode='human'):
        print("render() is not used")

    def get_info(self):
        return None

env = CartpoleRobot()
agent = PPOAgent(numberOfInputs=env.observation_space.shape[0], numberOfActorOutputs=env.action_space.n)
solved = False
episodeCount = 0
episodeLimit = 2000
# Run outer loop until the episodes limit is reached or the task is solved
while not solved and episodeCount < episodeLimit:
    observation = env.reset()  # Reset robot and get starting observation
    env.episodeScore = 0

    for step in range(env.stepsPerEpisode):
        # In training mode the agent samples from the probability distribution, naturally implementing exploration
        selectedAction, actionProb = agent.work(observation, type_="selectAction")
```



```

        # Step the supervisor to get the current selectedAction's reward, the new observation and whether we reached
        # the done condition
        newObservation, reward, done, info = env.step([selectedAction])

        # Save the current state transition in agent's memory
        trans = Transition(observation, selectedAction, actionProb, reward, newObservation)
        agent.storeTransition(trans)

        if done:
            # Save the episode's score
            env.episodeScoreList.append(env.episodeScore)
            agent.trainStep(batchSize=step)
            solved = env.solved()  # Check whether the task is solved
            break

        env.episodeScore += reward  # Accumulate episode reward
        observation = newObservation  # observation for next step is current step's newObservation

    print("Episode #", episodeCount, "score:", env.episodeScore)
    episodeCount += 1  # Increment episode counter

if not solved:
    print("Task is not solved, deploying agent for testing...")
elif solved:
    print("Task is solved, deploying agent for testing...")
observation = env.reset()
while True:
    selectedAction, actionProb = agent.work(observation, type_="selectActionMax")
    observation, _, _, _ = env.step([selectedAction])

```

10. Berikut hasil yang didapatkan


C:\Users\MSI GF63\Documents\Tutorial folder\cartPoleWorld\worlds\cartPoleWorld.wbt (cartPoleWorld) - Webots R2022a
File Edit View Simulation Build Overlays Tools Wizards Help
Simulation View
0:00:00:288 - 0.00x
scale 1 1 1
children
name "robot"
model ""
description ""
contactMaterial "default"
immersionProperties
boundingObject USE BODY
physics Physics
locked FALSE
translationStep 0.01
rotationStep 0.262
radarCrossSection 0
recognitionColors
controller "robotSupervisorController"
controllerArgs
customData ""
supervisor TRUE
Selection: controller (String)
robotSupervisorController
Select...
Edit
...s\Tutorial folder\cartPoleWorld\controllers\robotSupervisorController\robotSupervisorController.py
four_wheeled_collision_avoidance.py
robotSupervisorController.py
Console - All

```
from deepbots.supervisor.controllers.robot_supervisor import RobotSuper
from utilities import normalizeToRange, plotData
from PPO_agent import PPOAgent, Transition

from gym.spaces import Box, Discrete
import numpy as np


class CartpoleRobot(RobotSupervisor):
    def __init__(self):
        super().__init__()
        self.observation_space = Box(low=np.array([-0.4, -np.inf, -1.3,
                                     high=np.array([0.4, np.inf, 1.3, n
                                     dtype=np.float64)
        self.action_space = Discrete(2)

        self.robot = self.getSelf()  # Grab the robot reference from th
        self.positionSensor = self.getDevice("polePosSensor")
        self.positionSensor.enable(self.timestep)

        self.poleEndpoint = self.getFromDef("POLE_ENDPOINT")
        self.wheels = []
        for wheelName in ['wheel1', 'wheel2', 'wheel3', 'wheel4']:
            wheel = self.getDevice(wheelName)  # Get the wheel handle
            wheel.setPosition(float('inf'))  # Set starting position
            wheel.setVelocity(0.0)  # Zero out starting velocity
            self.wheels.append(wheel)

        self.stepsPerEpisode = 200  # Max number of steps per episode
        self.episodeScore = 0  # Score accumulated during an episode
        self.episodeScoreList = []  # A list to save all the episode sc

    def get_observations(self):
        # Position on x axis
```

```
INFO: robotSupervisorController: Starting controller: python.exe -u robotSupervisorController.py
Episode # 0 score: 33
Episode # 1 score: 24
Episode # 2 score: 46
Episode # 3 score: 30
Episode # 4 score: 41
Episode # 5 score: 29
Episode # 6 score: 35
Episode # 7 score: 23
```