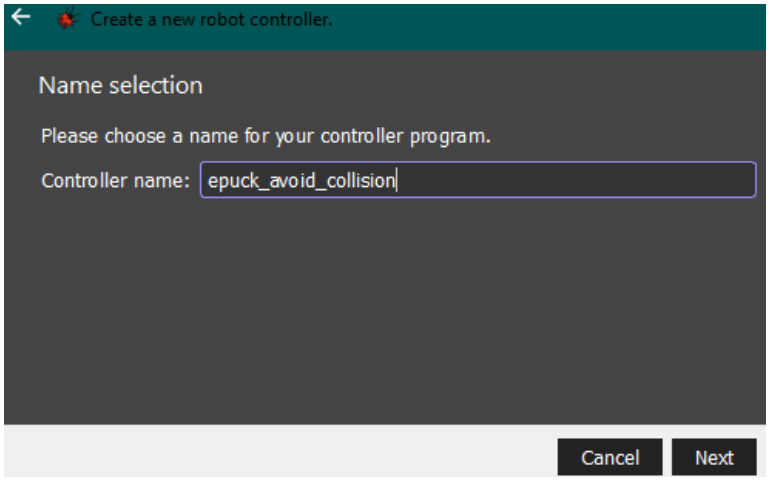
Hands-on 1



Hands-on 2

epuck_avoid_collision.py*

```python
"""epuck_avoid_collision controller."""

# You may need to import some classes of the controller module. Ex:
#  from controller import Robot, Motor, DistanceSensor
from controller import Robot, DistanceSensor, Motor

TIME_STEP = 64

# 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 order to get the
# 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 stopping the controller
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, like:
    #  motor.setPosition(10.0)
    pass

# Enter here exit cleanup code.

```

Hands-on 3 : Di Python tidak ada *main function*, program memulai eksekusinya dari awal file

epuck_avoid_collision.py*

```python
"""epuck_avoid_collision controller."""

# You may need to import some classes of the controller module. Ex:
#  from controller import Robot, Motor, DistanceSensor
from controller import Robot, DistanceSensor, Motor

TIME_STEP = 64

# 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 order to get the
# 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 stopping the controller
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, like:
    #  motor.setPosition(10.0)
    pass

# Enter here exit cleanup code.

```

Hands-on 4

epuck_avoid_collision.py*

```python
from controller import Robot, DistanceSensor, Motor

TIME_STEP = 64
MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# initialize devices
ps = []
psNames = [
    'ps0', 'ps1', 'ps2', 'ps3',
    'ps4', 'ps5', 'ps6', 'ps7'
]

for i in range(8):
    ps.append(robot.getDevice(psNames[i]))
    ps[i].enable(TIME_STEP)

leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))
leftMotor.setVelocity(0.0)
rightMotor.setVelocity(0.0)

# feedback loop: step simulation until receiving an exit event
while robot.step(TIME_STEP) != -1:
    # read sensors outputs
    psValues = []
    for i in range(8):
        psValues.append(ps[i].getValue())

# detect obstacles
right_obstacle = psValues[0] > 80.0 or psValues[1] > 80.0 or psValues[2] >
left_obstacle = psValues[5] > 80.0 or psValues[6] > 80.0 or psValues[7] >

# initialize motor speeds at 50% of MAX_SPEED.
leftSpeed  = 0.5 * MAX_SPEED
rightSpeed = 0.5 * MAX_SPEED
# modify speeds according to obstacles
if left_obstacle:
    # turn right
```