# Practical 1) Developing a World for your robot in Webots

Challenges, Build a suitable world for your robot which meets the following constraints.

1) A good-sized arena,
2) Solid walls or barriers to keep the robot within a set location.
3) Get your robot set up

## First Task: creating the project folder.

Use the wizard to create a new project folder.

Applied_AI_[Your Username]

Make sure to click add an arena.

Make sure to back up your project folder either on your one drive, USB or via email.

## Second Task: Changing Floor size

On the left panel, expand the Rectangle arena object.

Change the floor size to be equal to 10m2

## Third Task: Changing the floor colour

Delete the Floor appearance node. Your floor of your arena is now white and you will see a node saying floorAppearance NULL.

Double click on floorAppearance NULL and add the appearance base node

Double click on the texture node and add an image texture node

In your project folder, create a new folder called textures.

Your file structure should look like this

Webots\project\Applied_AI_Username\texture

In the texture folder, create a suitable png in any image editing software for your colour. For the time being this may just be a solid colour. Set the resolution of your image to 1024 for now. You can change this later if you wish.

Back in webots under the ImageTexture node, change the url node to point to your texture.

Check the arena floor matched your expectation, you can also try editing the image to see how it changes in webots.

Any errors relating to size: Resize your image in gimp/paint etc

SAVE! AND SAVE REGULARLY!

You can use the above steps to change the background of your environment

## Fourth Task: Adding some obstacles.

Minimize all node on the side panal.

Click the plus icon (default shortcut ctrl+shift+a)

Add a solid node

You can change the name of the solid in the dialog box below by entering a name in the def field. I.e., DEF: [Obstacle]

Open the solid node. Select children then add a shape.

Under the shape node firstly add a node to the the geometry (Box, Ball, Cone etc) to define the shape. The size of the shape can be adjusted using the X/Y/Z

add a texture as we did with the Floor.

Trying moving the object around using the arrows or shift left clicking

If the object ever gets lost you can set its position in the Translation giving x/y/z coordinates 0/0/0 being right in the centre.

SAVE!

Pause the simulator if it is running.

Move the object to centre 2 meters of the ground (0,2,0 in the translation).

SAVE!

Open the obstacle node up, double click on physics and add the physics node.

SAVE!

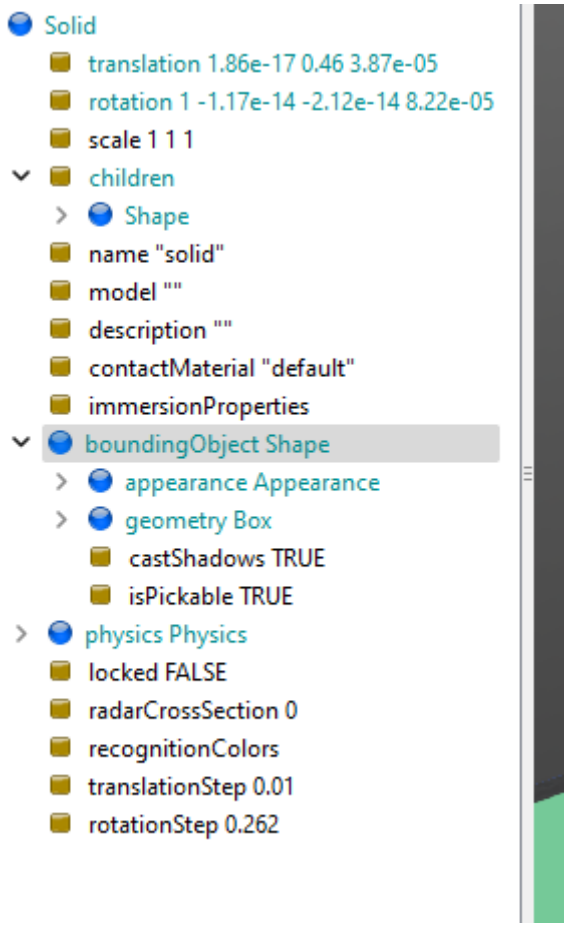
Push play on the simulator, did you object fall through the environment?

Pause the simulator,

Reset the world to it's last save point using the rewind icon or shortcut (ctrl+shit+t by default)

Open the obstacles children, and copy (ctrl + C) the shape node

Select the boundingObject node and paste the shape information here (ctrl + P). It should look like the following img




SAVE!

Start the simulation. Your object will now hit the ground.

The Shape node, sets the appearance of the object only.

The bounding object node sets the physical properties of the object.

If you change one, you must change the other.

Spend a bit of time adding a few more objects to your environment, try different shapes and physic settings. You have freedom to design the environment as you want it.

You may also want to have a look at preexisting nodes under the proto heading.

Once you have finished making your world, download the pionier.wbo from canvas. Add nodes-> import -> .wbo file

SAVE!!!

## Fifth Task: Adding a camera

Open up the Pioneer Node -> extension -> add a node -> camera

Note that the camera can be moved using the translation coordinates. I.e., you can move the camera forward and backwards

Lets make a new controller for the robot which we can edit.

Wizards -> new controller -> c -> gcc/makefile -> make sure open in editor is selected.

Copy and paste the code into the new file and save both the code (top right above the editor) and world (top left above the editor). If you cant see the original code it is also on canvas under unit 3

Back to the robot

Click controller -> select -> change this to the file you just made.

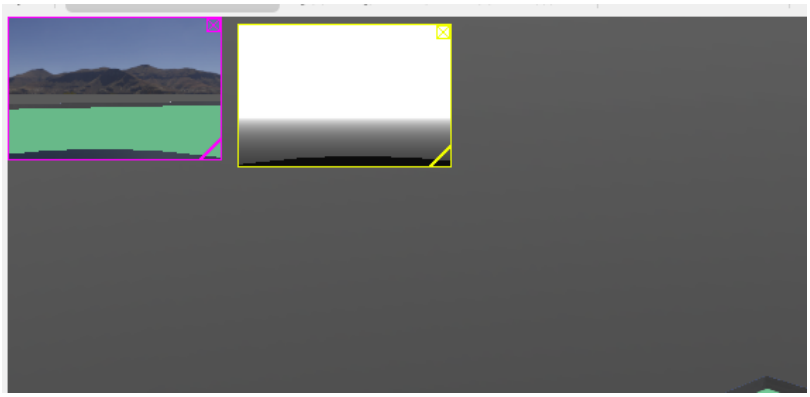
Six Task: Enable the camera

In the main() function add the following lines under the velocity. Around line 79

WbDeviceTag camera = wb_robot_get_device("camera");

wb_camera_enable(camera, time_step);

Your camera is now active, you may need to move Kinect camera (black and white camera) by dragging and dropping the window. You should see something along the lines of the following figure.



Let the robot run, it should be showing some default braitenberg behaviour.

SAVE! And back up!

Advanced: Work we will explore next week, but you can start having a go now to build up your skills.

Look through the code and make notes on how it works, what are the key variables and their roles.

Take the braitenberg behaviour out of the main loop and turn it into a function.

Experiment with adding some new behaviours to the robot