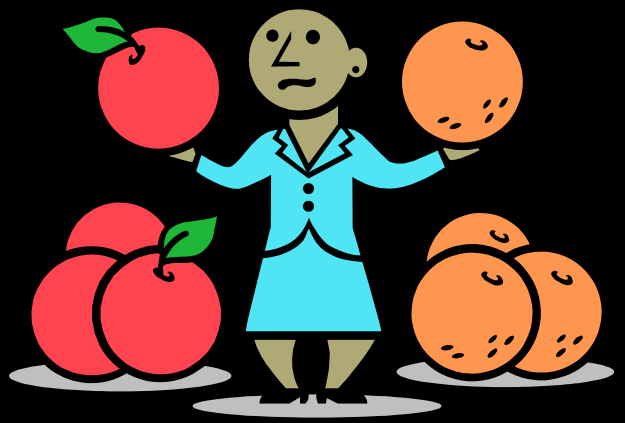
Webots Basics

Computer vs. Robotic Programs

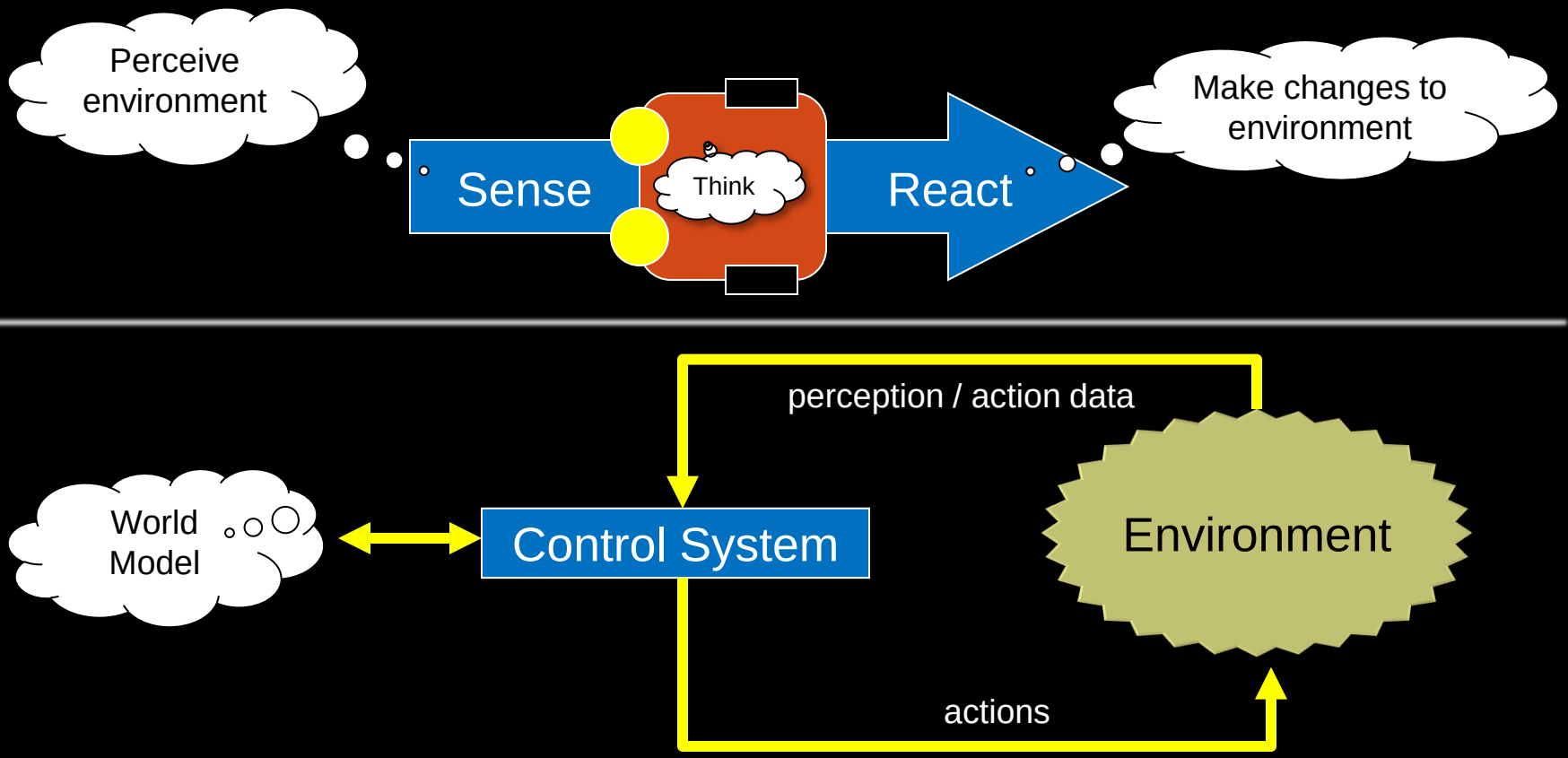
- **Computer Programs:**
  - designed to compute an answer
  - data usually valid when available
  - predictable program flow
  - foreseen errors easily handled



- **Robotic Programs:**
  - designed to react to achieve goals, not “an answer”
  - sensors often produce invalid data (or data missing)
  - unpredictable situations due to dynamic environment
    (e.g., unforeseen obstacles, wrong or missing sensor data, communication outages, hardware failure, etc..)
  - program must degrade gracefully in difficult situations

Robot Processing

- Here is the basic “flow of control” commonly used:



Robot Control - Pseudocode

```
MyRobotController() {
    Set up sensors and motors
    WHILE (TRUE) {
        detectLeft = read left sensor;
        detectRight = read right sensor;

        IF (detectLeft) THEN
            turn = right;
        ELSE IF (detectRight) THEN
            turn = left;
        ELSE
            turn = none;

        IF (turn == left) THEN
            turn leftMotor on backwards
            turn rightMotor on forward
        ELSE IF (turn == right) THEN
            turn rightMotor on backwards
            turn leftMotor on forward
        ELSE
            turn rightMotor on forward
            turn leftMotor on forward
    }
}
```

Each robot program runs in an infinite loop.

Initialize
Set up the motors, sensors, variables etc...

Sense
Read sensors to determine if a collision is detected on the left or right sensor.

Think
Decide whether robot should go straight or turn away.

React
Turn left, turn right, or move forward accordingly.

Robot Control – JAVA code

Informs JAVA compiler where to find libraries of various functions that you will use in your program. You will add lots more **import** statements.

All classes are given a name which must match the filename … (**LabController.java** in this case)

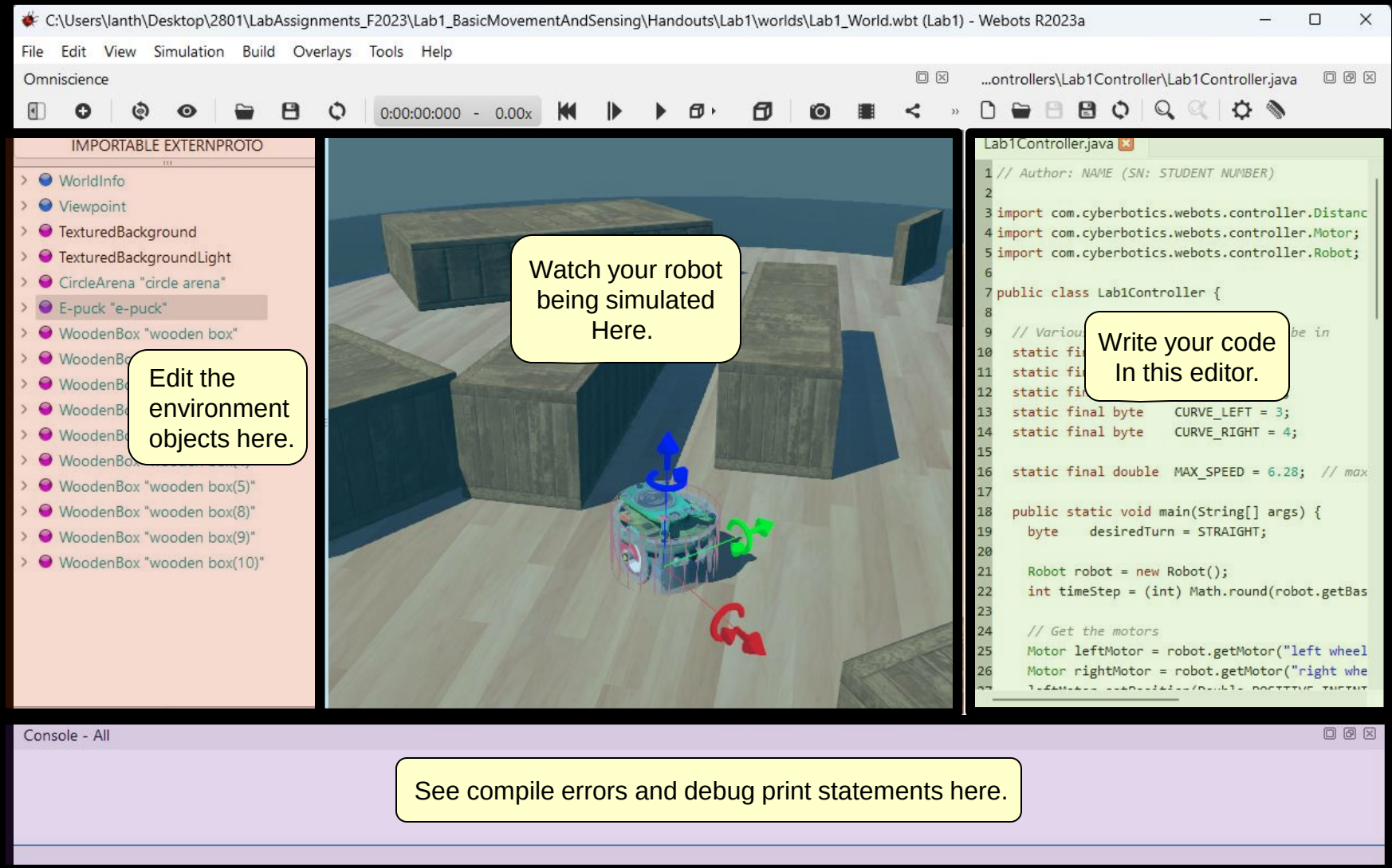
```
import com.cyberbotics.webots.controller.Robot;

public class LabController {
   /* Declare static constants & variables here */
   public static void main(String[] args) {
      Robot robot = new Robot();          // Creates a generic Robot object
      int timeStep = (int) Math.round(robot.getBasicTimeStep());

      /* INITIALIZE … variables, sensors, motors, etc… */

      while(robot.step(timeStep) != -1) {

          /* SENSE … Read sensors */

          /* THINK … Make decision as to what to do */

          /* REACT … Move motors, head, arms, etc… */

      }
   }
}
```

main function is starting point of your program.

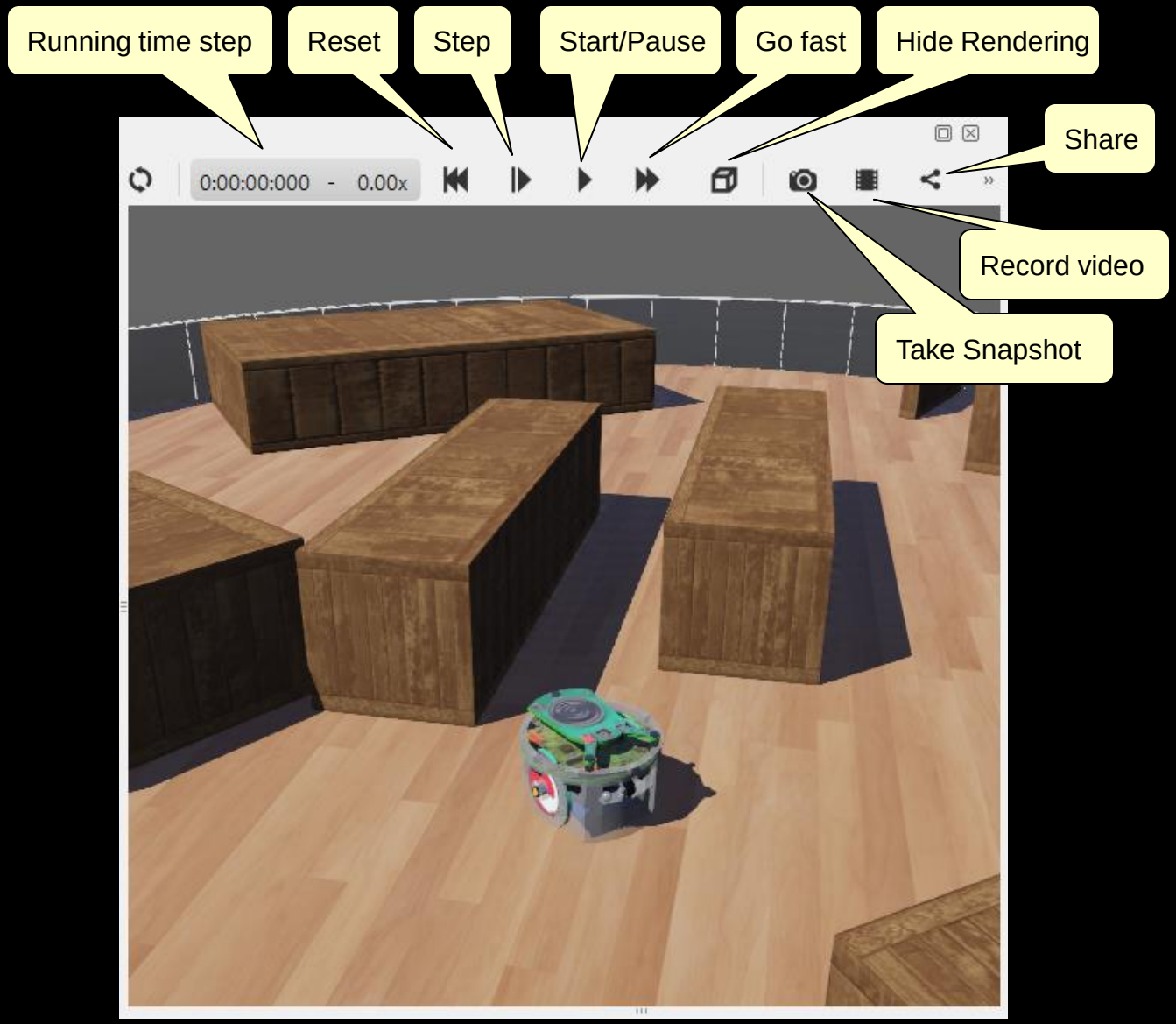
Simulation time-step in (ms). You won't change this.

Loops forever (until PAUSED or STOPPED)

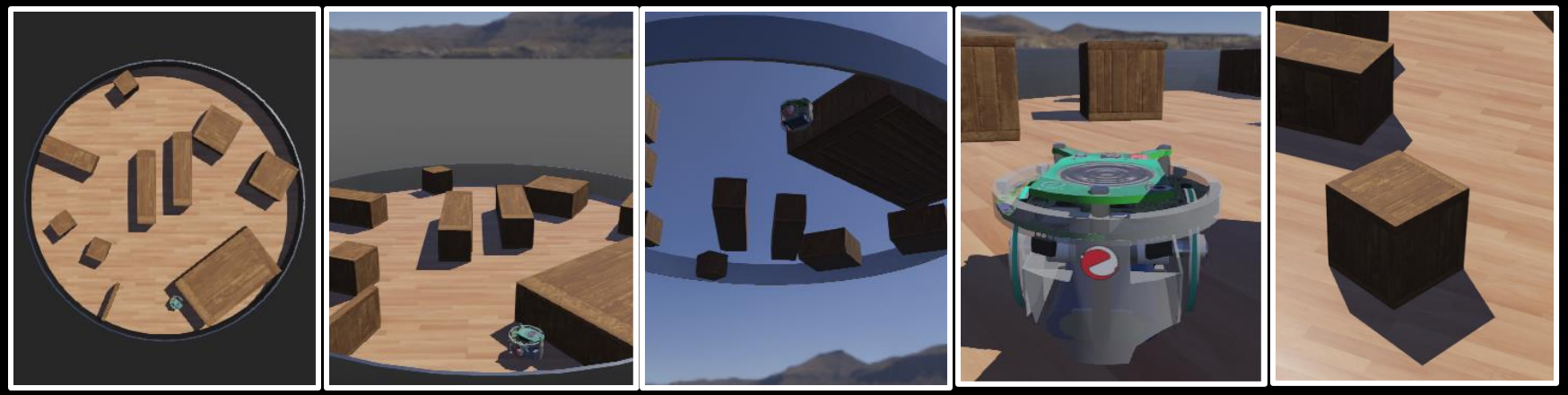
Webots Interface – main GUI



Webots Interface – Simulation Controls

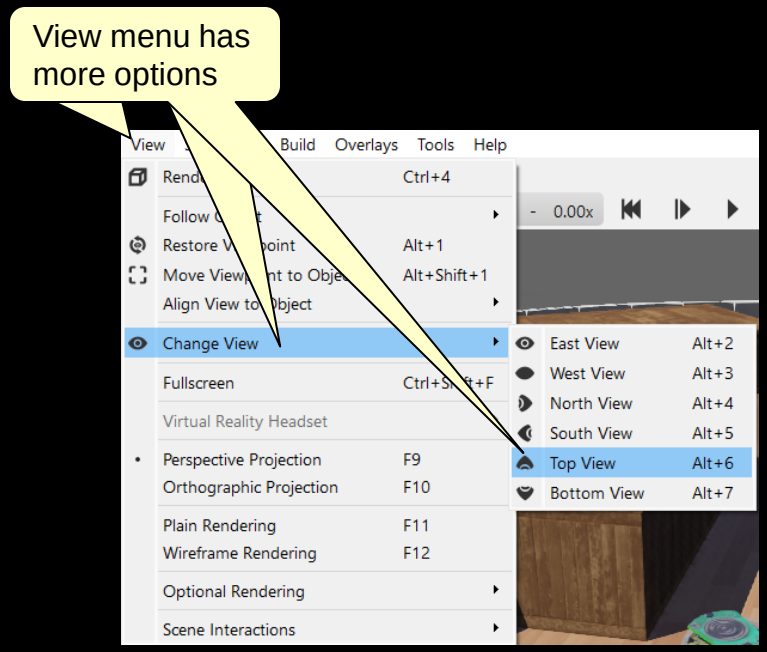


Webots Interface – Changing Viewpoint

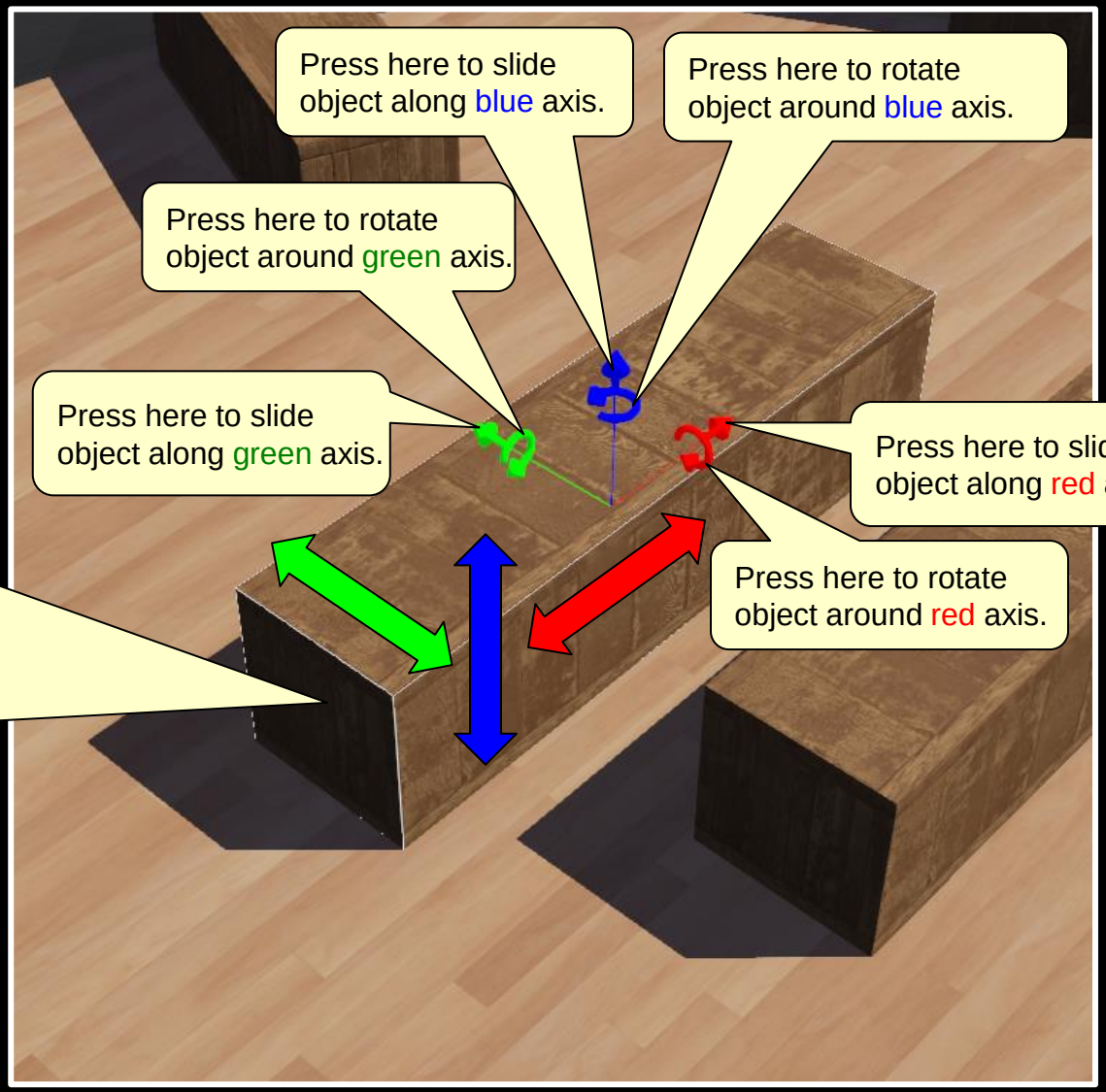


Use the mouse to change viewpoint:

Roll: left press + up/down
Spin: left press + left/right
Translate: right press + up/down/left/right
Zoom: scroll up/down
Select: left click
Menu: right click



Webots Interface – Moving World Objects

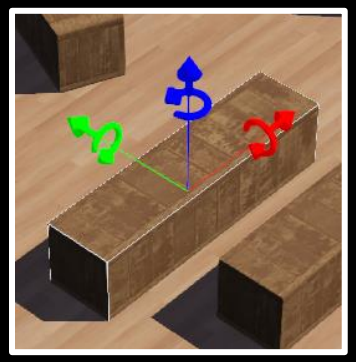


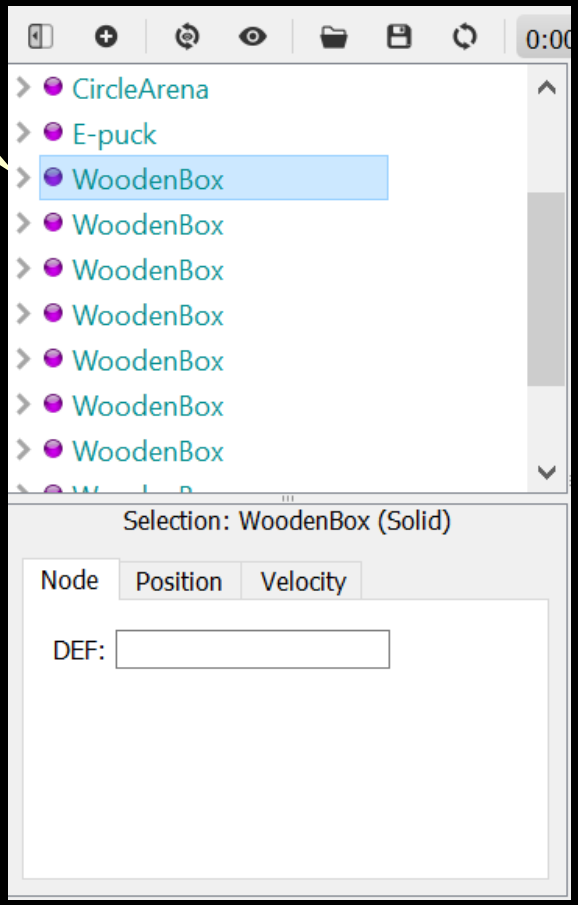
Click anywhere on object to select it, then press and hold left mouse button on one of 6 places to change object.

Webots Interface – Editing World Objects

1. Click on the **>** to expand or collapse one object's attribute view.

Objects are all listed here in the **Scene Tree**.

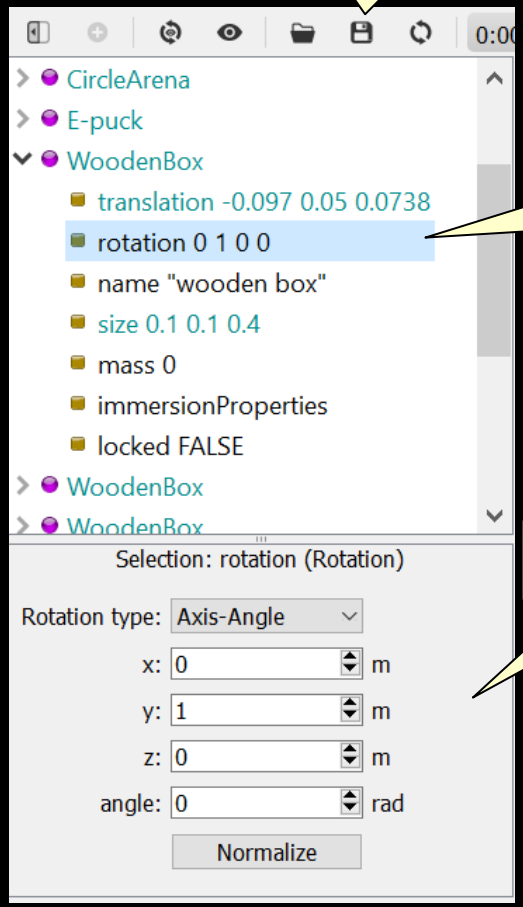




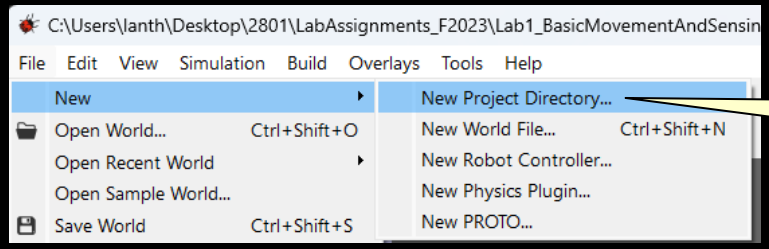
4. Save your changes.

2. Select an attribute.

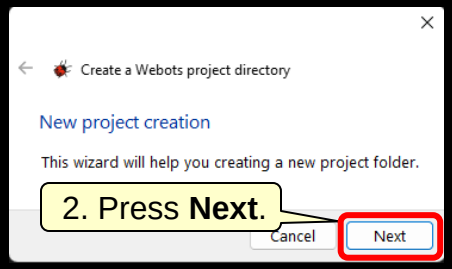
3. Change its values.



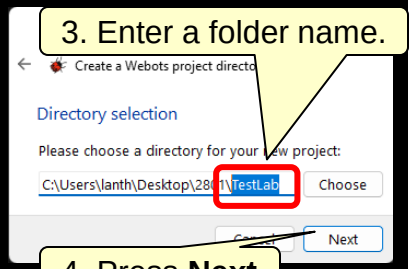
Webots Interface – Making New World



1. Select **New Project Directory**… from the **File/New** menu



2. Press **Next**.



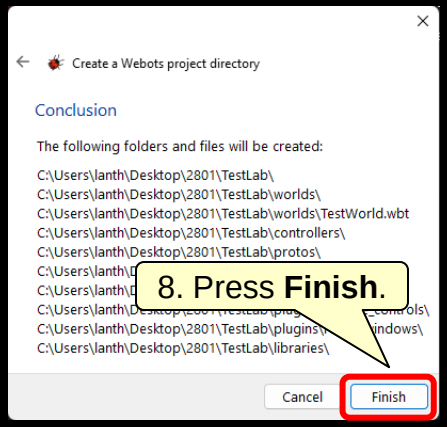
3. Enter a folder name.

4. Press **Next**.

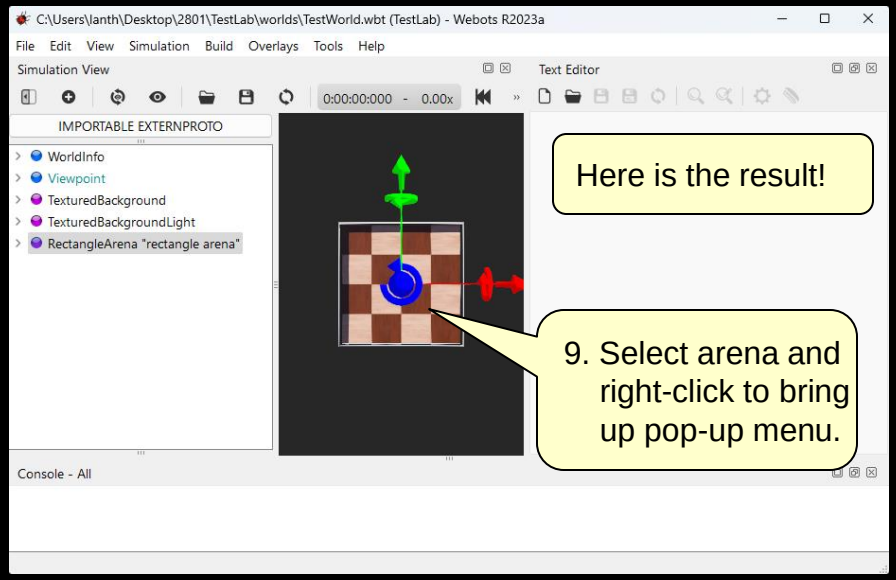
5. Enter name for world.

6. Check this.

7. Press **Next**.



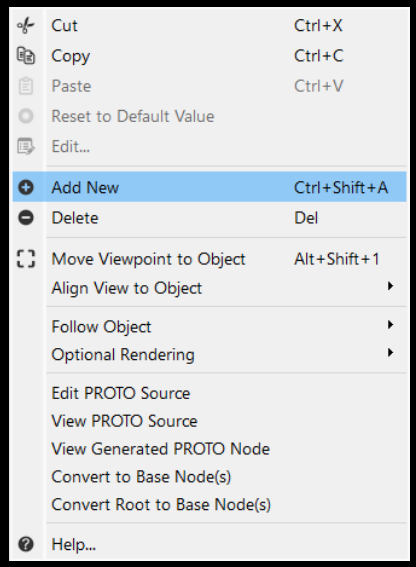
8. Press **Finish**.



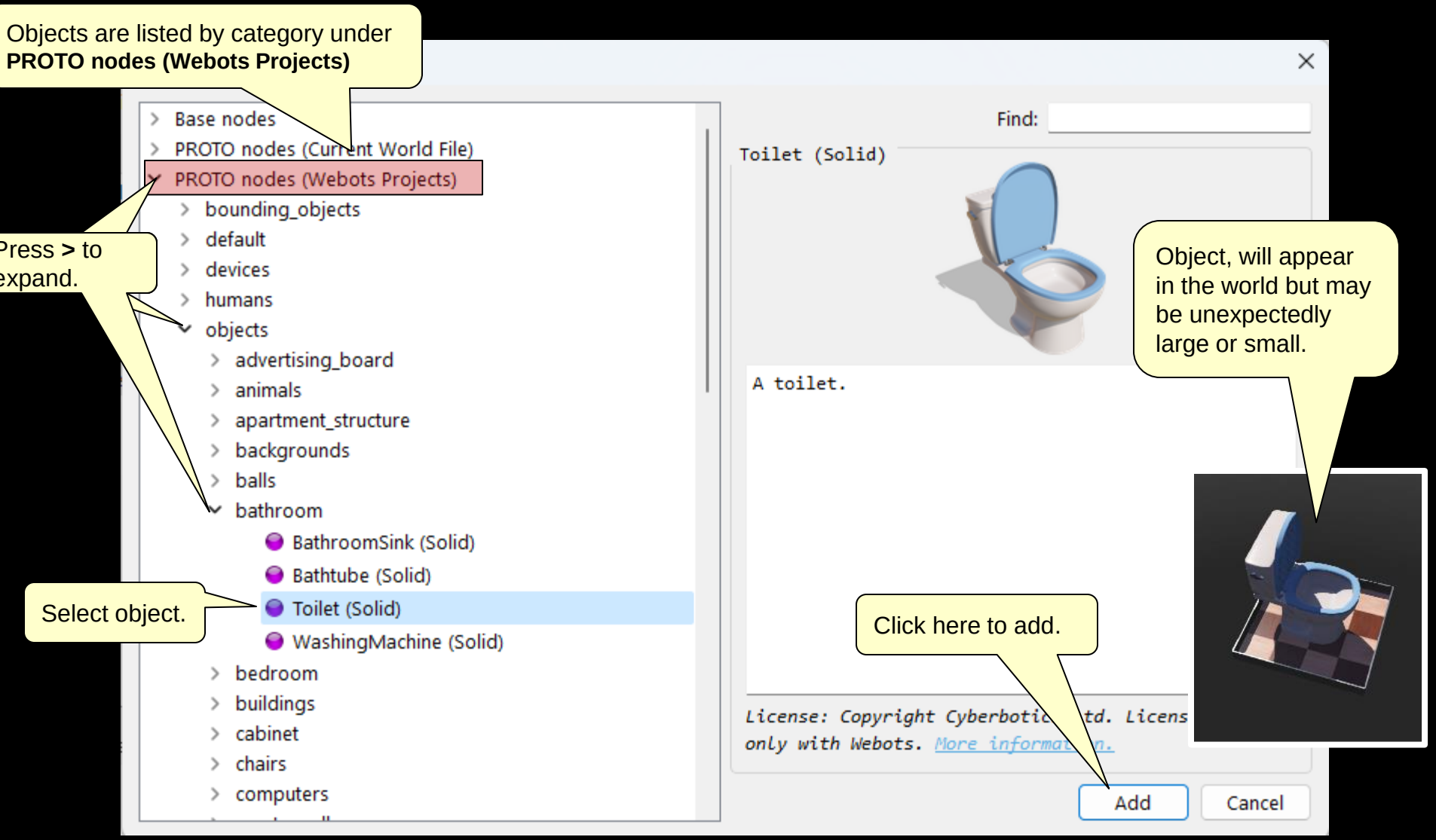
Here is the result!

9. Select arena and right-click to bring up pop-up menu.

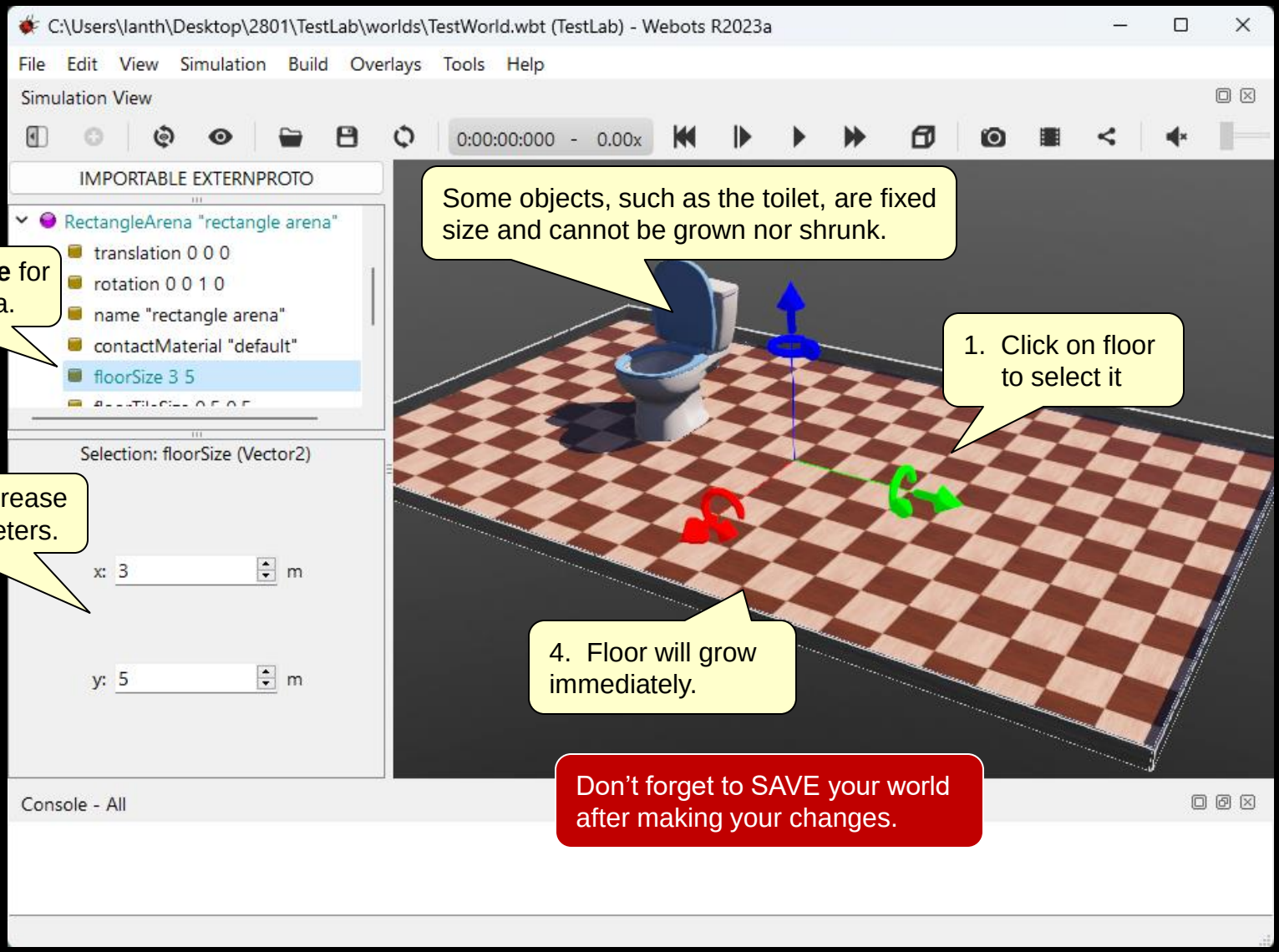
10. Select **Add New** to add objects and robots. The menu can take up to 10 seconds to appear!



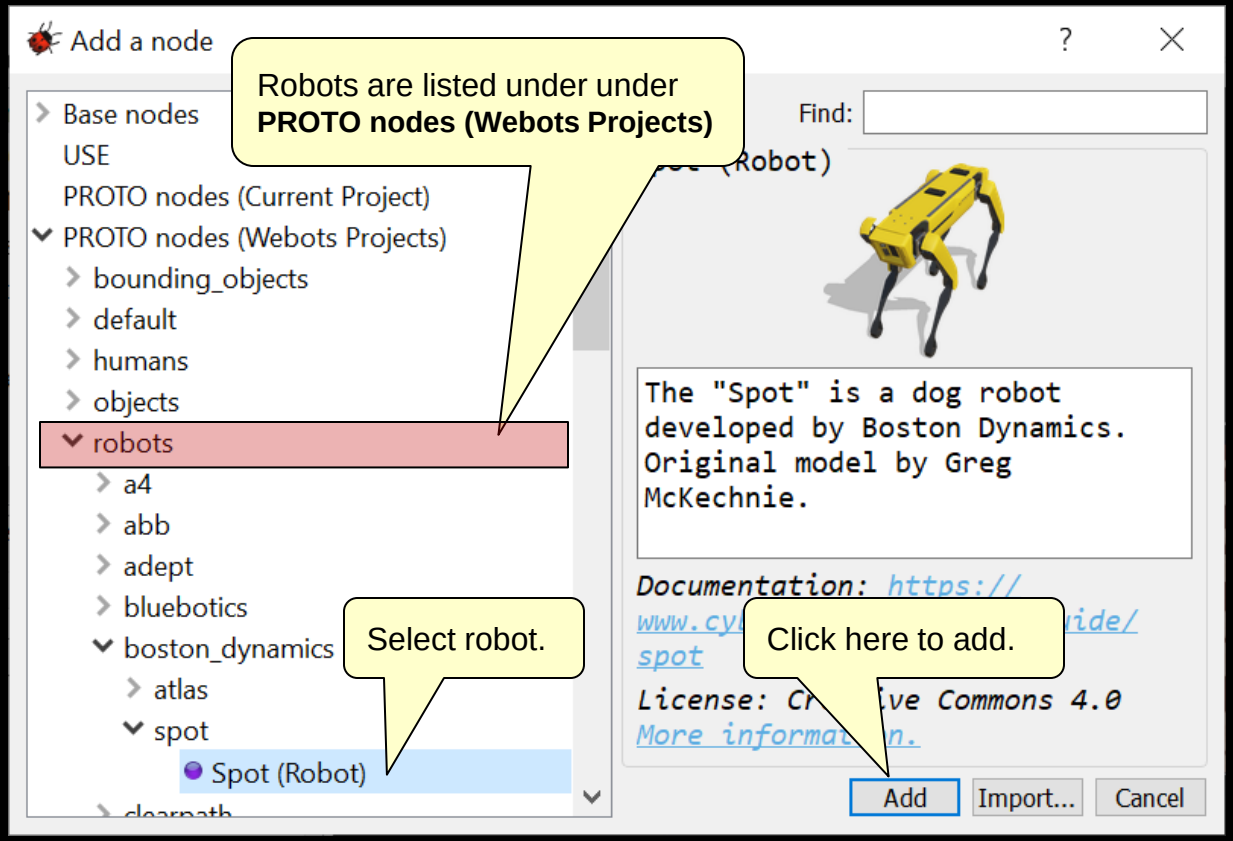
Webots Interface – Adding an object

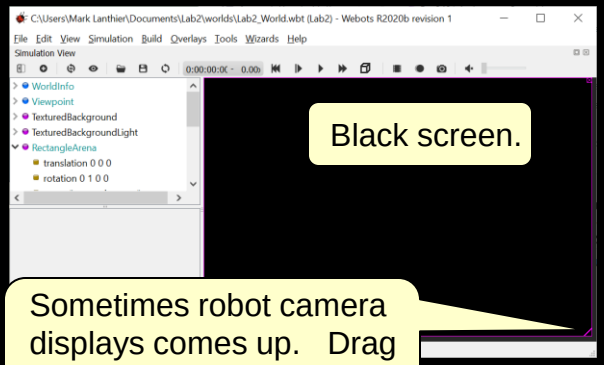


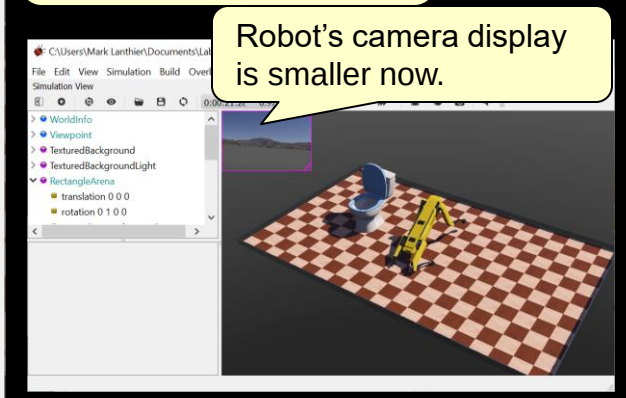
Webots Interface – Resizing an Object



Webots Interface – Adding a Robot

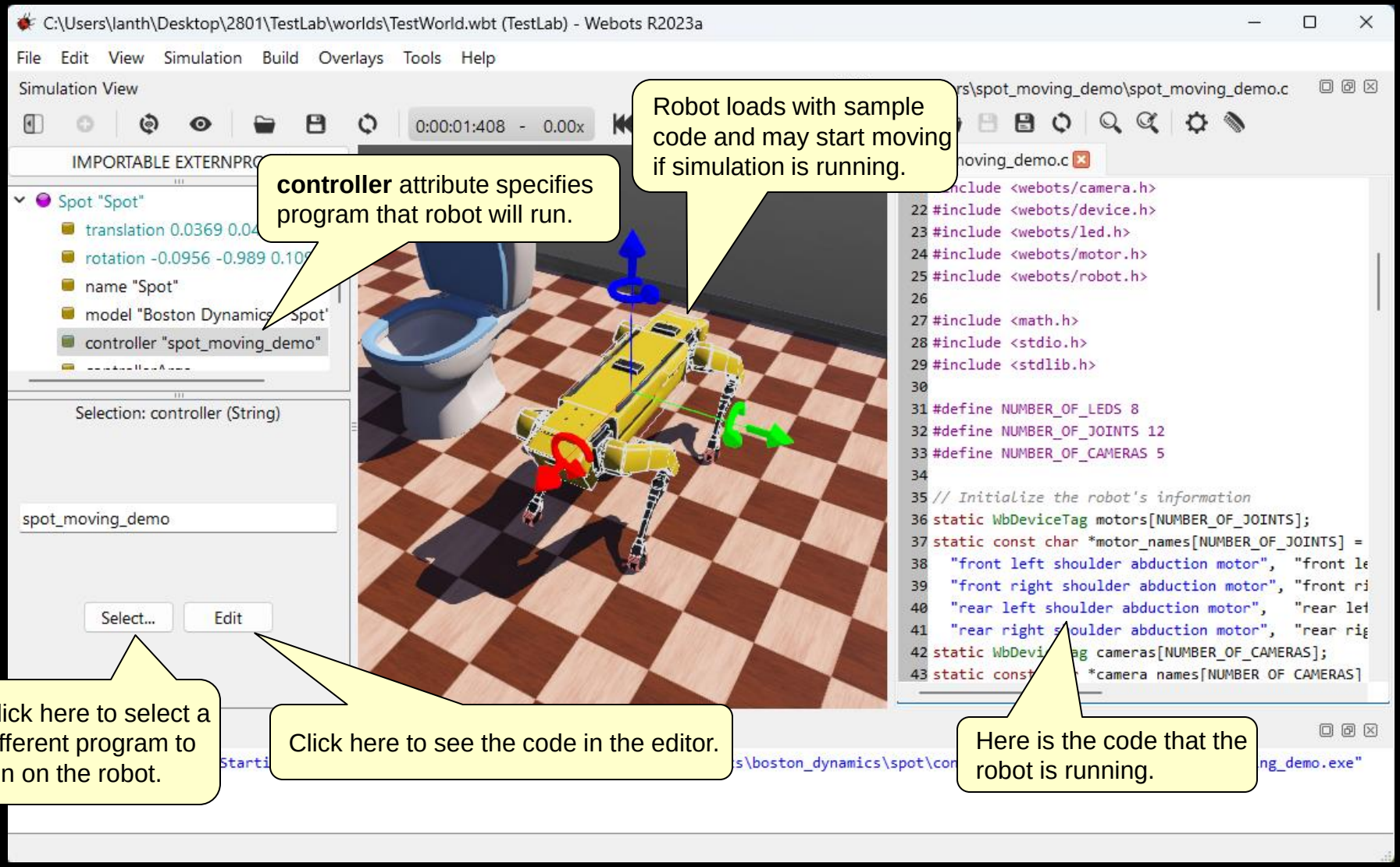




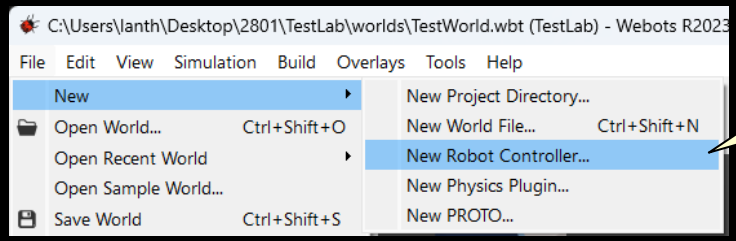


Don't forget to SAVE your world after adding your robot.

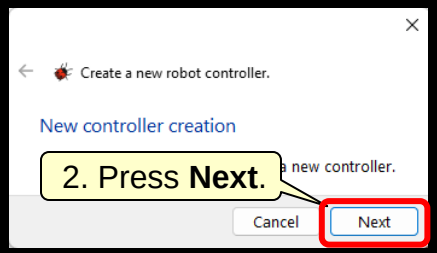
Webots Interface – Controller Code



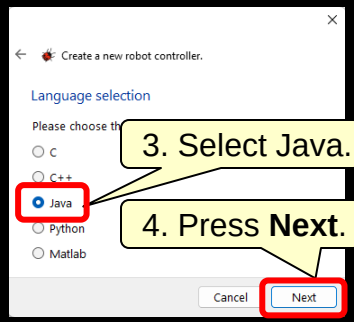
Webots Interface – Your Controller



1. Select **New Robot Controller**… from the **Feil/New** menu

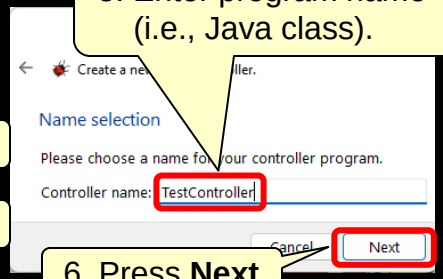


2. Press **Next**.



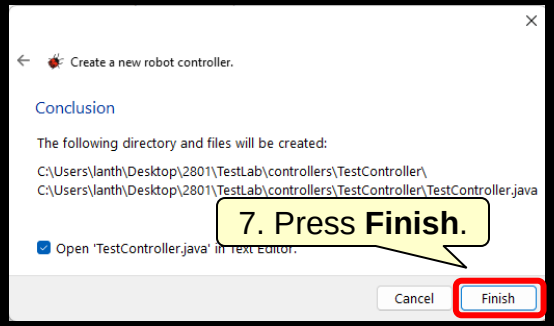
3. Select Java.

4. Press **Next**.



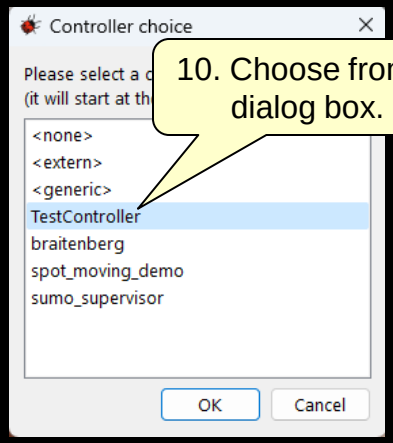
5. Enter program name (i.e., Java class).

6. Press **Next**.

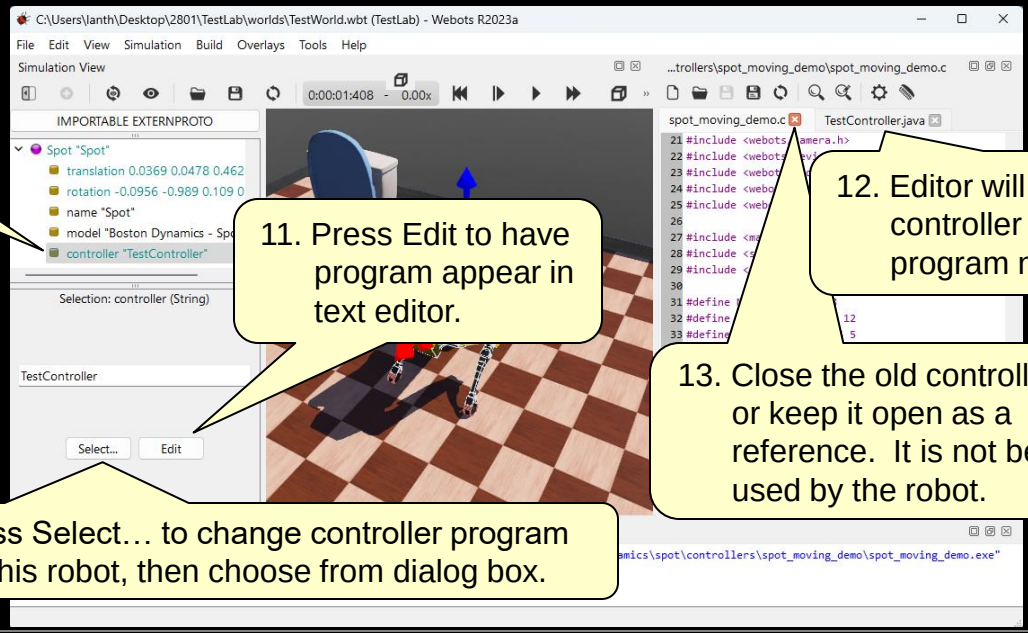


7. Press **Finish**.

8. Select controller attribute for this robot.



10. Choose from dialog box.



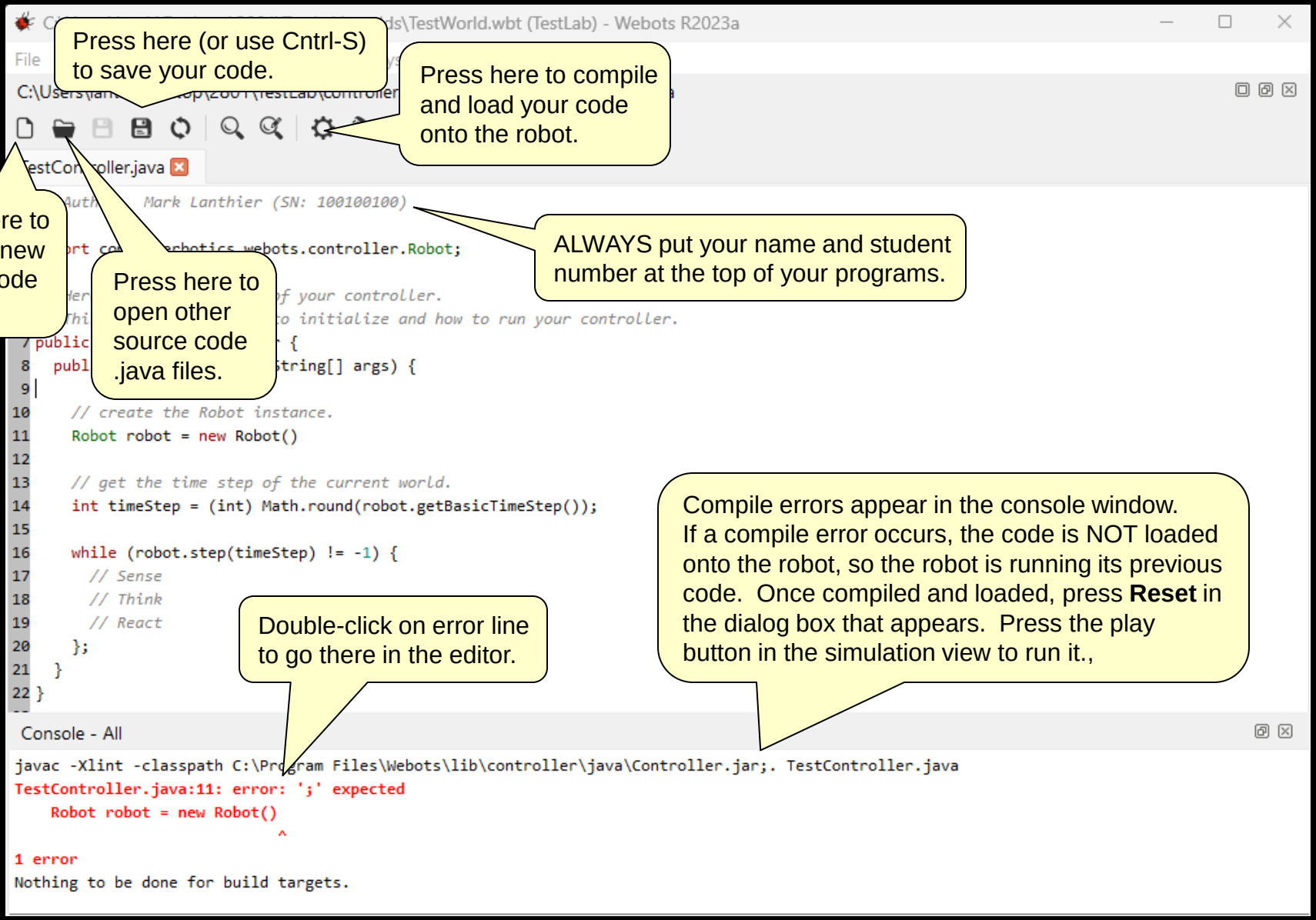
11. Press Edit to have program appear in text editor.

12. Editor will show a new controller template program now.

13. Close the old controller, or keep it open as a reference. It is not being used by the robot.

9. Press Select… to change controller program for this robot, then choose from dialog box.

Webots Interface – Editing Code



Start the Lab …