TUTORIAL: MoveYourRobot with Unity3D

You created your own robot with servo-motors and you are wondering how to control it.

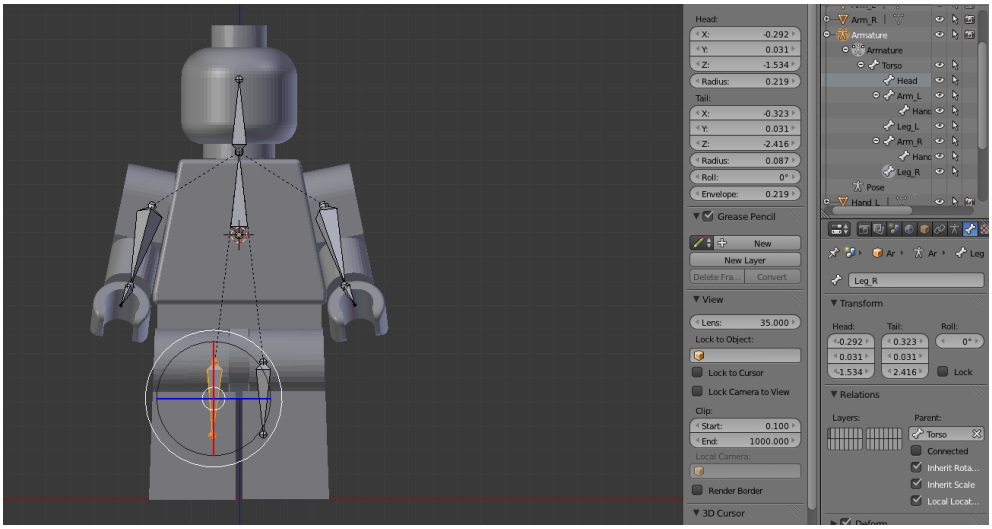
This package provide environment and scripts to be easily able to control your robot thank to unity, with animations, speeches, tracking, speech recognition…

I will make this tutorial with a robot LEGO who have 4 DOF and a webcam on his head.

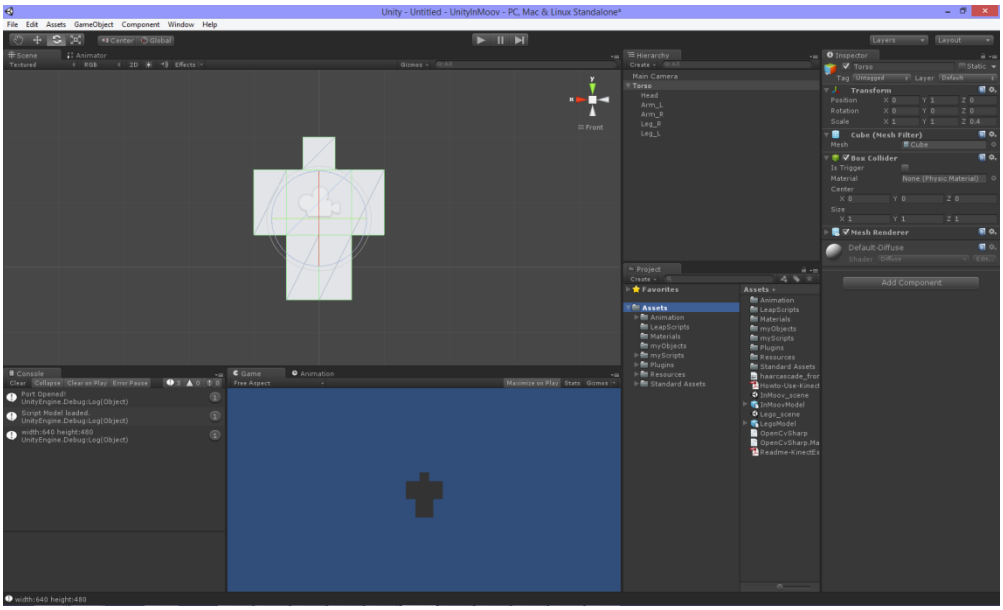
RIUS: Robot Interaction Unity System

Create Your Model

- You can use a 3D modeling software, like Blender (free), to have a realistic model. Realize your model and add an armature corresponding to enabled movements.
  There are many tutorials on YouTube like this one:
  https://www.youtube.com/watch?v=B9iL1hmJXrI



- Or you can use Unity and create quick model of your robot creating Cube, and drag'n'drop to link parenting.



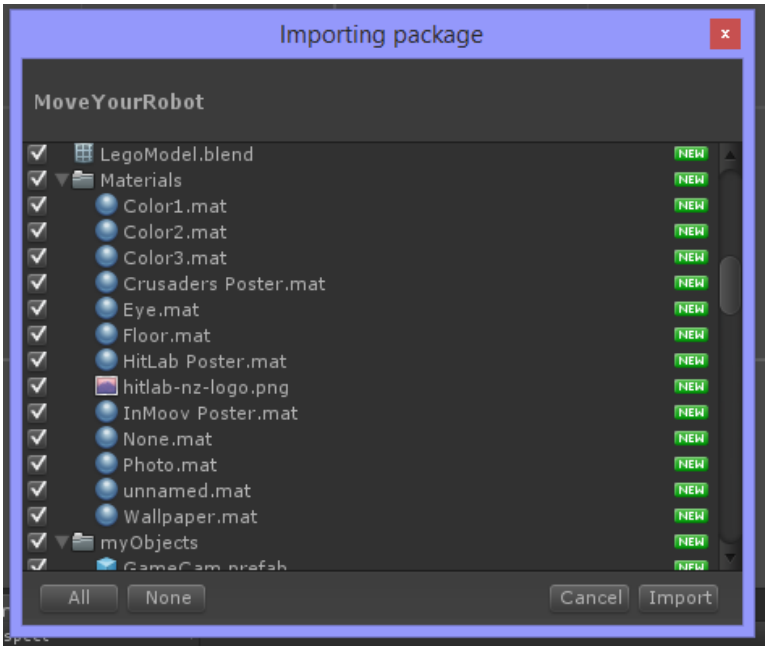
Set Your Unity Project

Launch Unity3D, open a new project.

(If you never used Unity3D before, you should first become familiar with the Unity interface, there are many tutorials on YouTube)

Download MoveYourRobot package, open a new Unity project and import the package:

Assets->Import Packages…



Let all the checkbox checked by default.

Import the model file, (.blend for Blender) : Assets -> Import New Assets

The package provides already two models, the InMoovModel and the LegoModel, in the Asset folder.

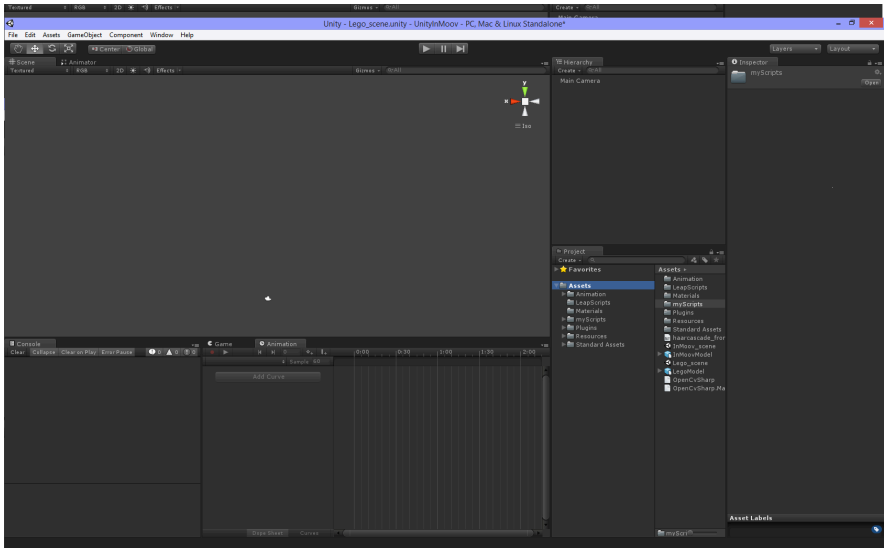
Go in File->Build Settings-> Player Settings ,

and then in the inspector, set Api Compability Level to ".NET 2.0"

Save the scene as MyRobot_scene (or what you want), or create a new one File->New Scene

The package provides already two example scene, the InMoov_scene and the Lego_scene, in the Asset folder.

You should have something like this :



Drag'n'Drop your model inside the scene.

Remove the main cam object.

Add Assets->MyObject->Room in the scene and set the position to (0,0,0).

Place your model in the center of the scene.

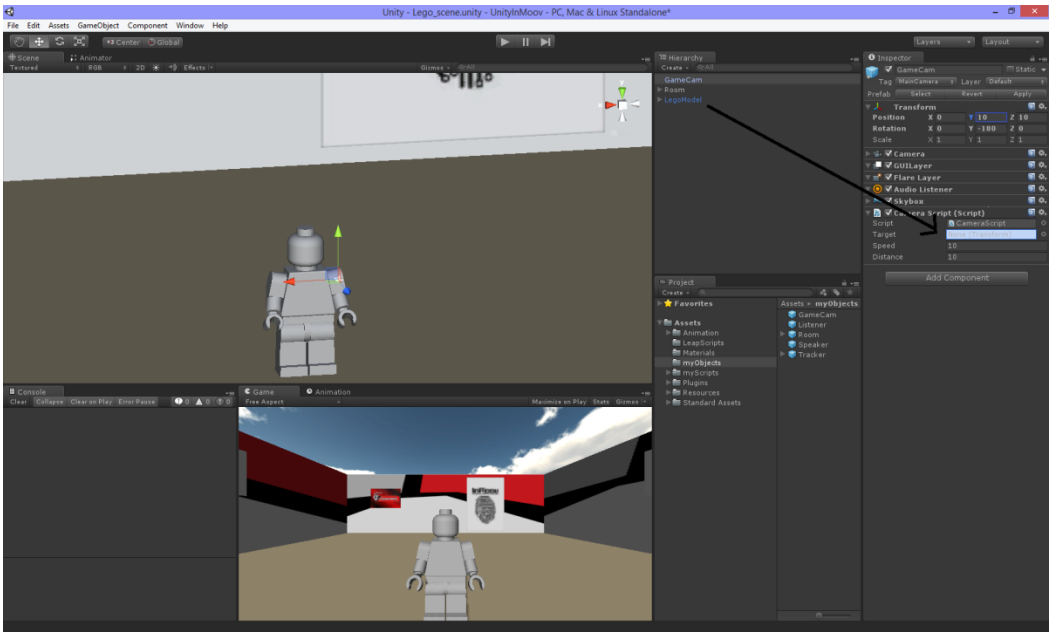
Rescale the model in the inspector to a normal scale.

Click on the model

In the Inspector -> Transform: Change the values of scale regarding to each axis.

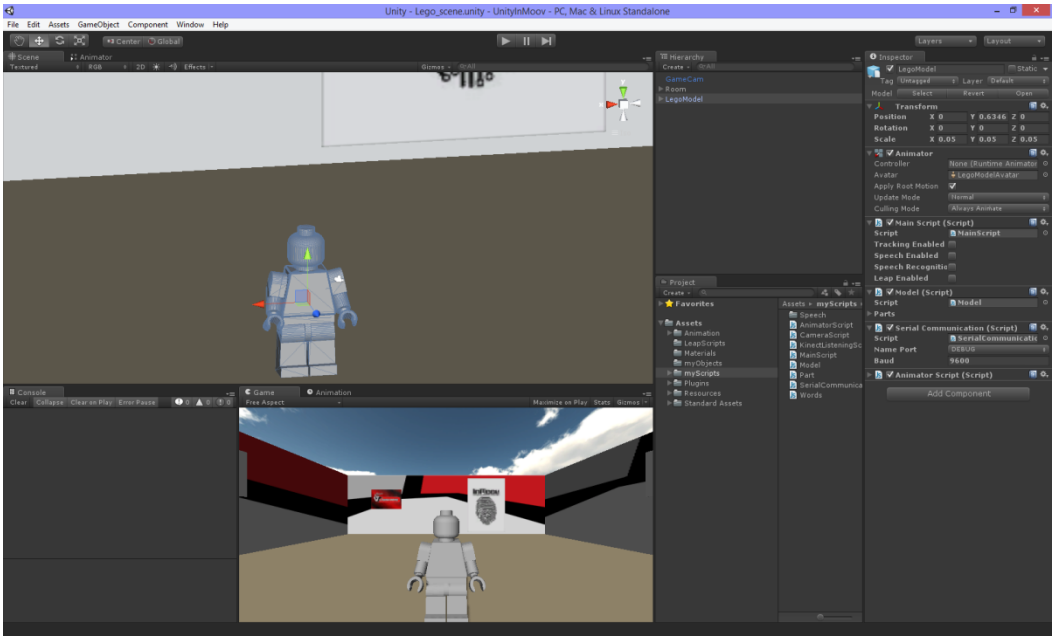
Add Assets->MyObject->GameCamera drag'n'drop inside the scene and set the position to have a good vision of your model in the game window.

In the "CameraScript" attached to the "GameCamera" object, set your model as "Target" by drag'n'drop.



Add all the basics scripts to your model:

- MainScript
- Model
- SerialCommunication
- AnimatorScript



Then you have to set the parameters of the Model script:

Click on your Model -> Inspector -> Model component:

Click on Parts :

Size = the numbers of servo-motors you want to control.

Name = Name of your rotative parts (head, Arm_L…).

PinId = the pin number corresponding with your Arduino Board.

Pieces = the pieces which are moving with the servo (default is just 1 but it could have more than one pieces, in fingers for example…). We will see it in the Arduino part of this tutorial.

Bone : drag'n'drop the Transform of your model corresponding to this parts.

Develop the hierarchy of your model's armature in the hierarchy window until have all the parts visible. Click back on your model, drag'n'drop the part from your model hierarchy to the Model scrip.

Min : the minimal angle in Euler degree(could be negative).

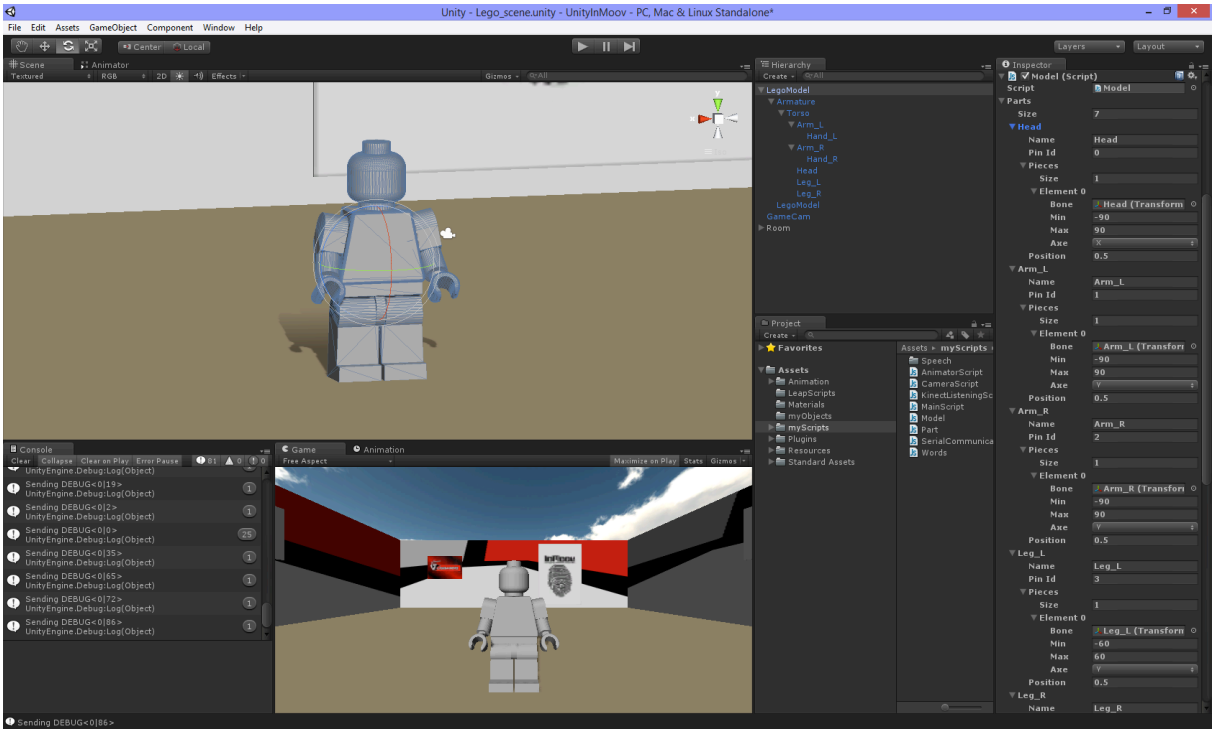
Max : the maximal angle in Euler degree.

Axe : the axis of rotation, mX = -X .

Position : is the initial position of the part

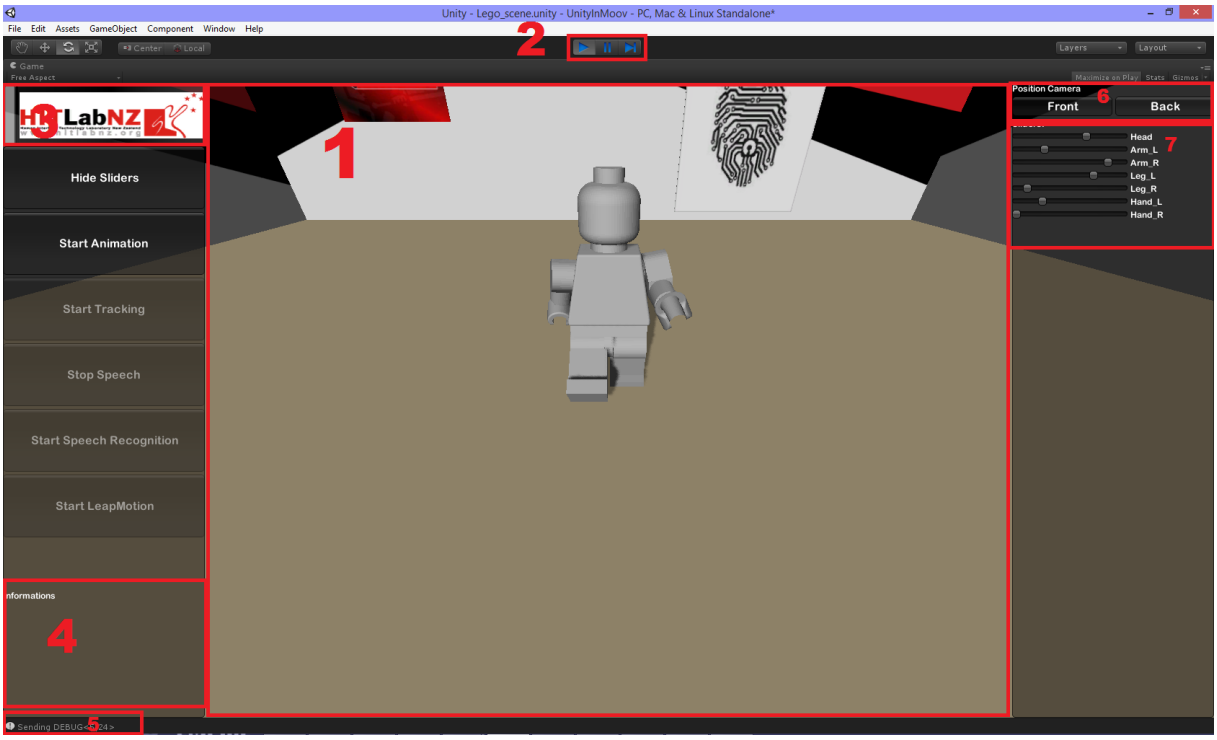
You have to experiment couple time to find the right parameters Min/Max/Axe.

We can’t exactly know the right parameters so you have to try.



Play Your Scene

You are now ready to control your Model, Click on Play, you should have something like this:



1: The scene, show the model and the environment.

2: Play/Stop : To quit click on play.

3: Click to Show/Hide Menus

4: Display information about contents

5: Show the debug log

6: Set the camera position in front or back of the model, you can also use the arrows keys or the right click to rotate the camera around the model. Use the mouse wheel to zoom.

7: Sliders which represents parts positions, use this to control your model and your robot.

Set Your Arduino Board

Connect the Arduino Board. To connect several motors, you need an external 5V power supply. You can take a battery or a computer power supply for example. Plug all your servos on the power supply (you can make it with a strip-board like on the Figure 4). You can use an Arduino Uno, or Mega, or you can combine several boards if you have many servos.

The most important part is connecting the ground of your Arduino boards to the minus of your power supply. You can damage your motors if you don't do that. The yellow wires represent the motor position signal. You can plug it in all the PWM and Digital pin of your board. Look into the Arduino software to know which port of your computer is connected to the Arduino Board. Then set the NamePort of the SerialCommunication script.

Use DEBUG if you don’t want to connect your robot yet, you will see the data sent in the debug log.

Create ANIMATIONS

There is many way to create animations on Unity, look the tutorial on YouTube to handle the AnimatorController of Unity.

In Animation, be care to rotate your parts only on their DOF, else the model animation will not correspond to your robot animation.

Animator Controller:

Create a new Animator Controller “MyRobotAnimator”

Click on your model; drag the new animator on the Animator->Controller

Animations:

I make the difference between two kinds of animation,

The “Posture”: its animations which represent one state of the robot, idle, T-pose…

You make animation by combining different postures; the transitions are doing the movements. This is the faster way to create basics animations.

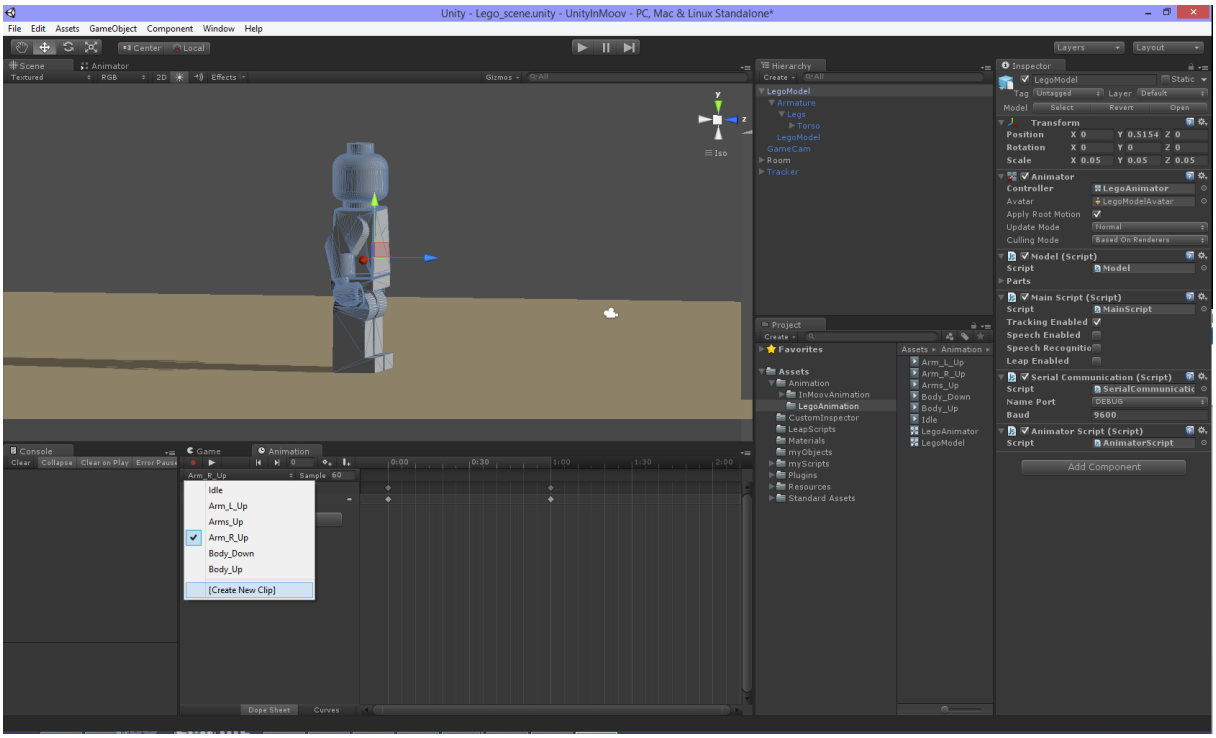
The “Motion”: it’s real animations with movements, speed, time…

It’s useful to create complicate and accurate animations.

In the Animation window

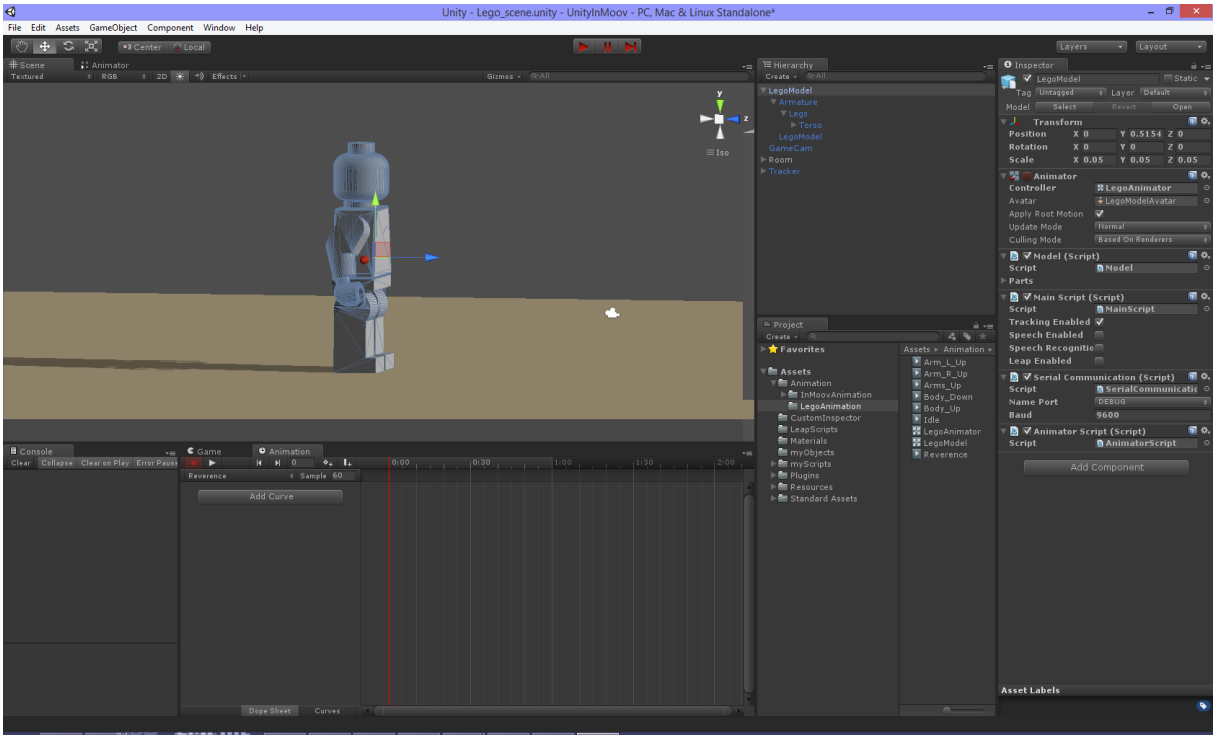
If you can’t create new animation, click once on your model.

[Create New Clip]



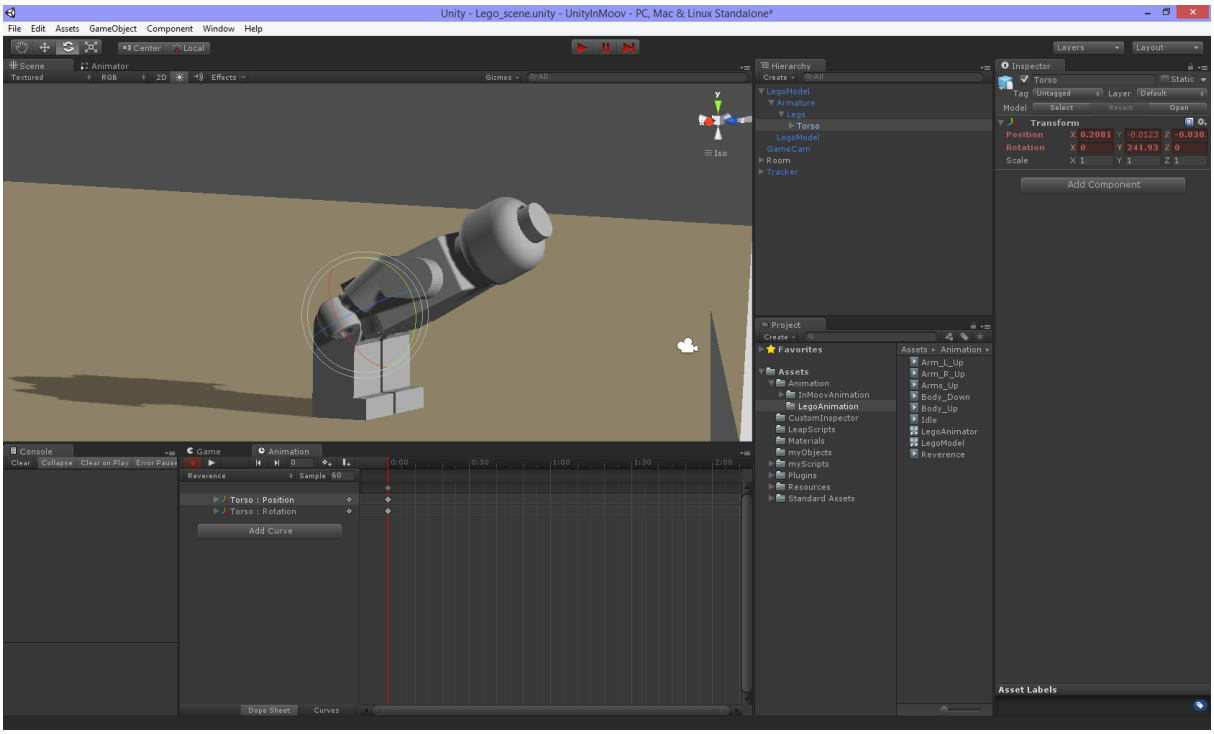
Posture:

Click at the beginning of the time, a vertical red line appears.



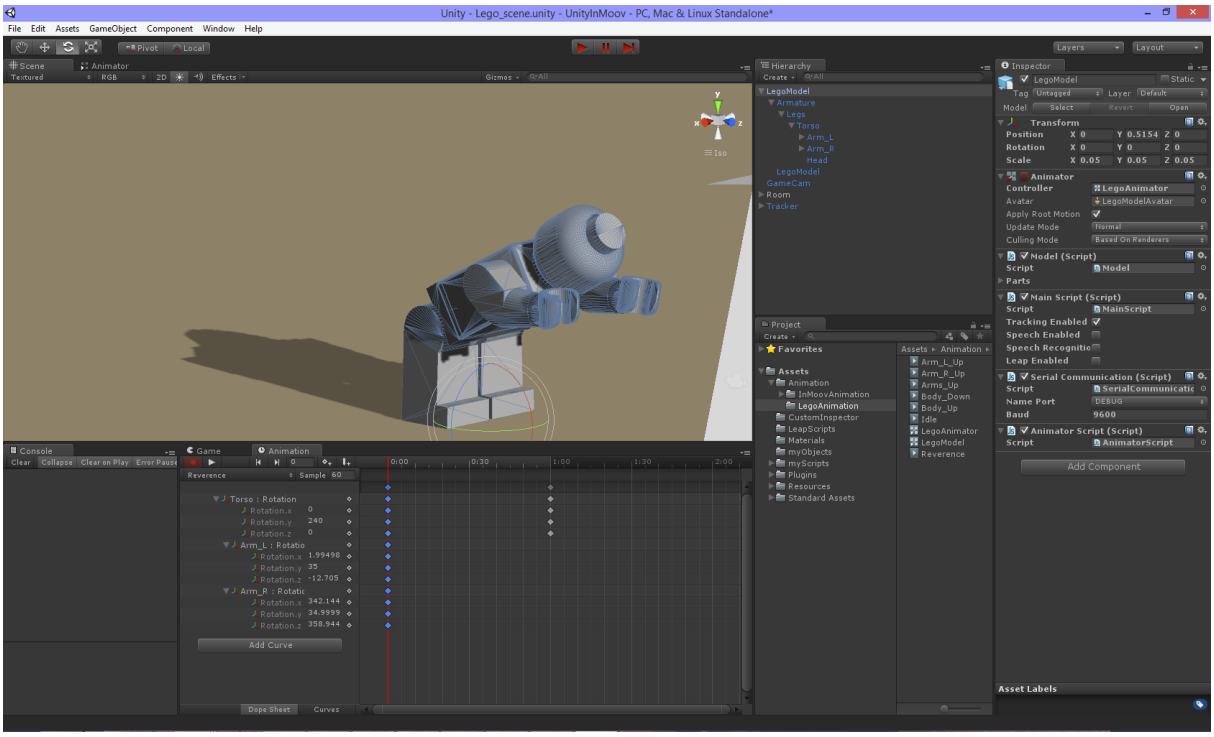
Then on the Hierarchy window, click on the part you want to move and in the scene window, rotate it a bit on the direction you want to.

On the Animation window the part's name should appears with position and rotation components, you can delete the position component.



Write the desired rotation on the correct axis; don't touch to the others rotations. Be care to stay in the interval of your min and max regarding to this part.

Repeat it for all the parts you want to rotate in this posture.



When all the part are set, click on the first blue square in the top of the vertical red line, it should select all the square, then copy Ctrl+C, put the red line at "1.00" and paste Ctrl+V.

Now you have an animation (Posture) which doing no movement but which describe a posture of your model.

Motion:

You can also create complexes animations like dances… Do the same thing but insert different position in different time.

Make a pack of basics animations for your robot.

Now we are ready to use the animator controller to combine yours animations and then program the behavior of your robot.

In the Animator window (different from Animation window !):

Drag animations from Assets->Animation->MyRobotAnimation to the Animator window

You can rename the animation. On the top of the inspector window.

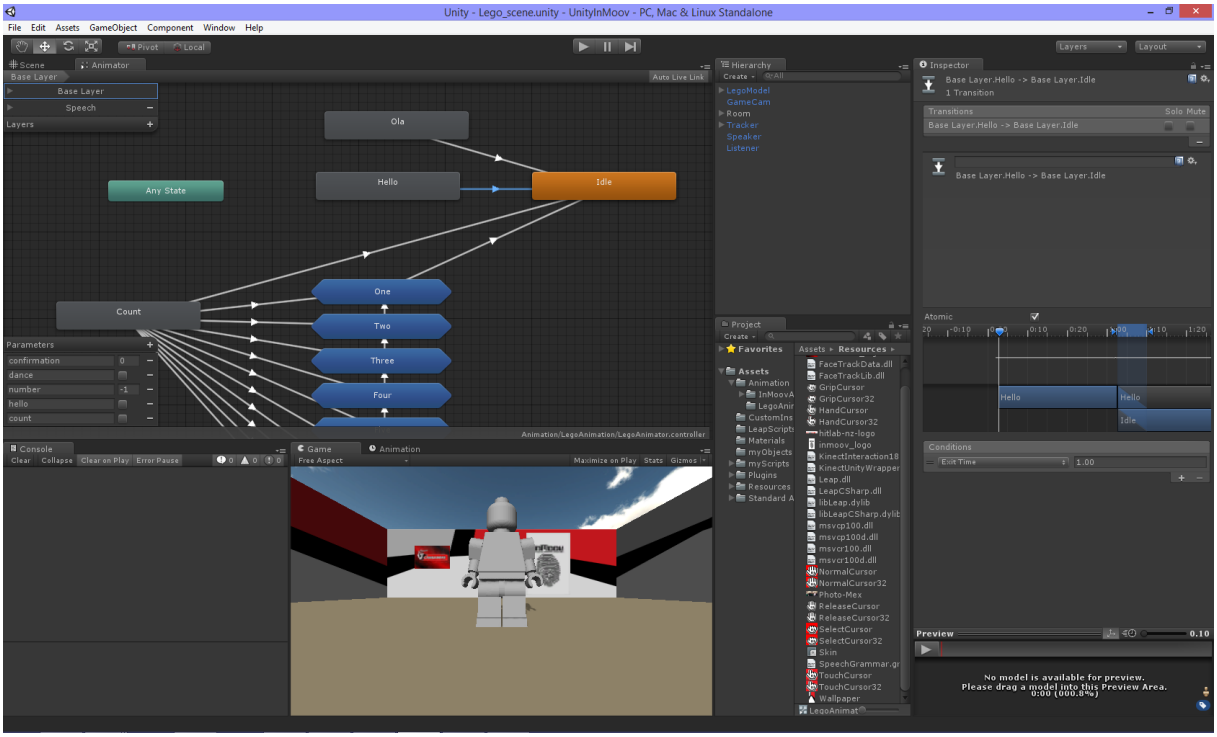
Right click on the animation you want to be play first and select "Set As Default".

Then you have to create transitions between your state (posture or motion).

Right click on a state, "Set Transition" and link it to another state.

Then you have to set the parameters of this transition, this is how you create the motion.

Click on the transition, look in the inspector.



If the state which the transition is from is a posture, then you can control the duration of the posture by set the Conditions parameter, “Exit Time” is the number of time the state will execute until doing the transition, because the posture duration is 1s if you follow the previous step, then this parameters will be the duration of your state in second.

If the state which the transition is from is a motion, set the Conditions parameter to the number time you want to execute this motion (1 by default). You can set the speed of the motion by clicking on it in the Animator window and set the Speed parameter.

If you make a transition between 2 postures, you can set the speed of the movement by moving the duration of the transition (the blue horizontal bar).

This is just the basics.

The Unity animation tool is so powerful; there are many tutorials on YouTube to learn how to use it. You can create very complex behavior.

Module TRACKING

Plug the robot camera to your computer, install the needed drivers…

Add the object Tracker to the scene: Assets->myObject->Tracker

In the Inspector:

Tracking Script: XTransform = drag the part of your robot which will rotate horizontally

YTransform = drag the part of your robot which will rotate vertically

If you just have one DoF for the Tracking, you can drag the Tracker object in the empty parameter.

Speed = speed of the movement (60 by default; if the speed it's too high it will be a jerky movement)

Accuracy = accuracy of the target (5% by default; too high -> the robot will not really target you; too low -> the robot will never be stable)

Capture Analyzer Script:

Index Cam: index of your robot camera on your computer (on a laptop, 0 is the integrated camera)

Flip Cam: check it if your camera is on the wrong direction.

On the hierarchy window, develop select Tracker->ViewScreen

ScreenScript:

Target: Drag your model

XTransform = drag the part of your robot which will rotate horizontally

YTransform = drag the part of your robot which will rotate vertically

If you just have one DoF for the Tracking, you can drag the Tracker object in the empty parameter.

Now in the MainScript, you can check the TrackingEnabled checkbox.

Module SPEECH

Be sure that your speaker is working.

I'm using the Windows voice, sorry for others users.

You have to place the dll file Assets->Voice_speaker.dll and put it in your system folder.

Windows 32B >> Copy 'Voice_speaker.dll' in windows\system32 folder

Windows 64B >> Copy 'Voice_speaker.dll' in windows\SysWOW64 folder

Add Assets->myObjects->Speaker to scene.

In inspector window:

VoiceSpeaker:

Voice_nb: set the index of the Windows voice you would use.

Add Assets->myScript->Speech->Speech.js to your model.

Now in the MainScript, you can check the SpeechEnabled checkbox.

Now to use the text-to-speech module:

The module is working accordingly with the AnimatorController.

Click on a state which you want to add a speech, and in the inspector window, write "Speech" in the Tag parameter to signal that this animation have a speech attached.

Then click on your model and in the Speech component, add an element to the Speech array by increase the size.

Enter the name of the animation.

Enter the text.

That's all !

Module SPEECHRECOGNITION

The more efficient way to make speech recognition is using a Kinect.

Install your Kinect and the SDK:

http://www.microsoft.com/en-us/download/details.aspx?id=43661

Be sure she is working by testing the Microsoft example (outside Unity).

Add Assets->myObject->Listener to the scene.

In the inspector:

KinectListeningScript:

Model : drag your model.

Normally you don't have to change the parameters of the SpeechManager script.

The Grammar File Name corresponds to the file .grxml which contains the recognized words.

Click on your model and add Assets->myScript->Words.js

Now in the MainScript, you can check the SpeechEnabled checkbox.

There already some keywords program:

YES, NO, START TRACKING, STOP TRACKING, START ANIMATION, STOP ANIMATION, SHOW SLIDERS, HIDE SLIDERS, HELLO, DANCE, COUNT, and numbers from ZERO to TEN.

The responses of the keywords detection are programmable in the words.js script.

For example, STOP TRACKING deactivate the tracking in MainScript, SHOW SLIDERS activate the sliders in MainScript.

But it's more useful to control the behavior of your robot.

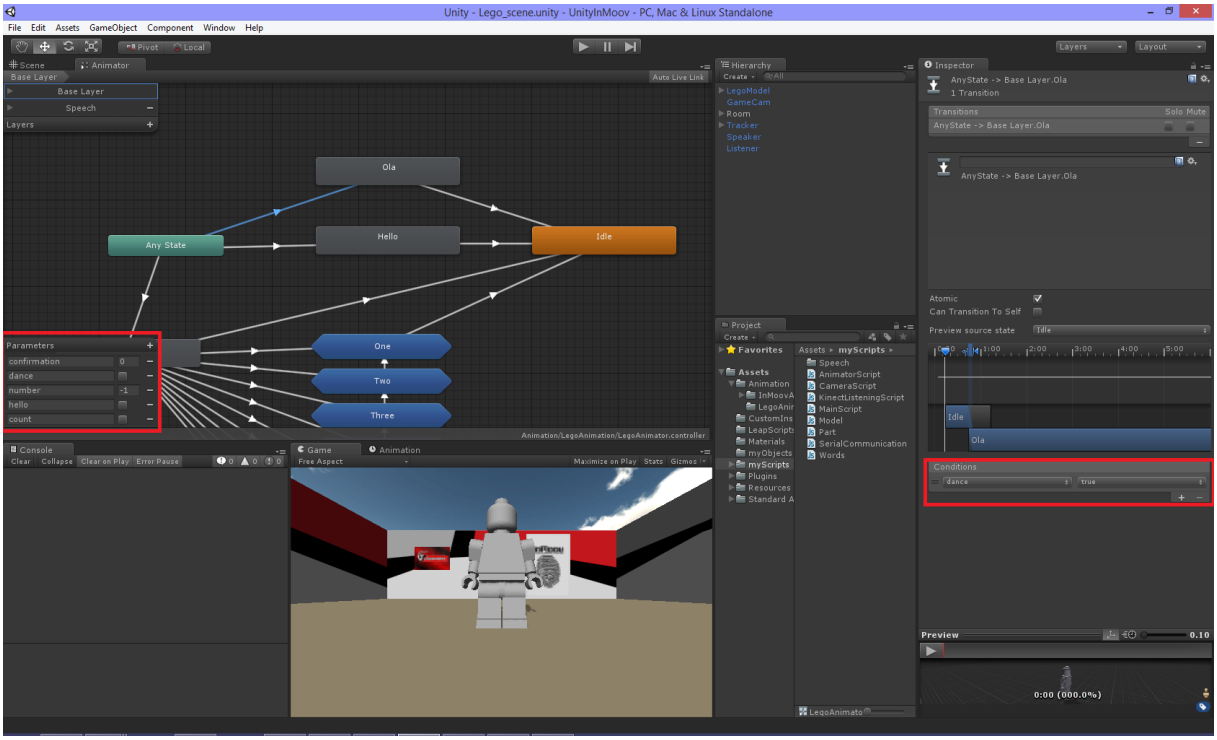
To do it we use parameters in the Animator window. There are different parameters type, we use Integer and Boolean.

For example the integer parameter "confirmation" allows us to deal with the Yes/No response.

-1 = No / 1 = Yes / 0 = Default

You can use these parameters as transitions conditions to navigate between yours different states.

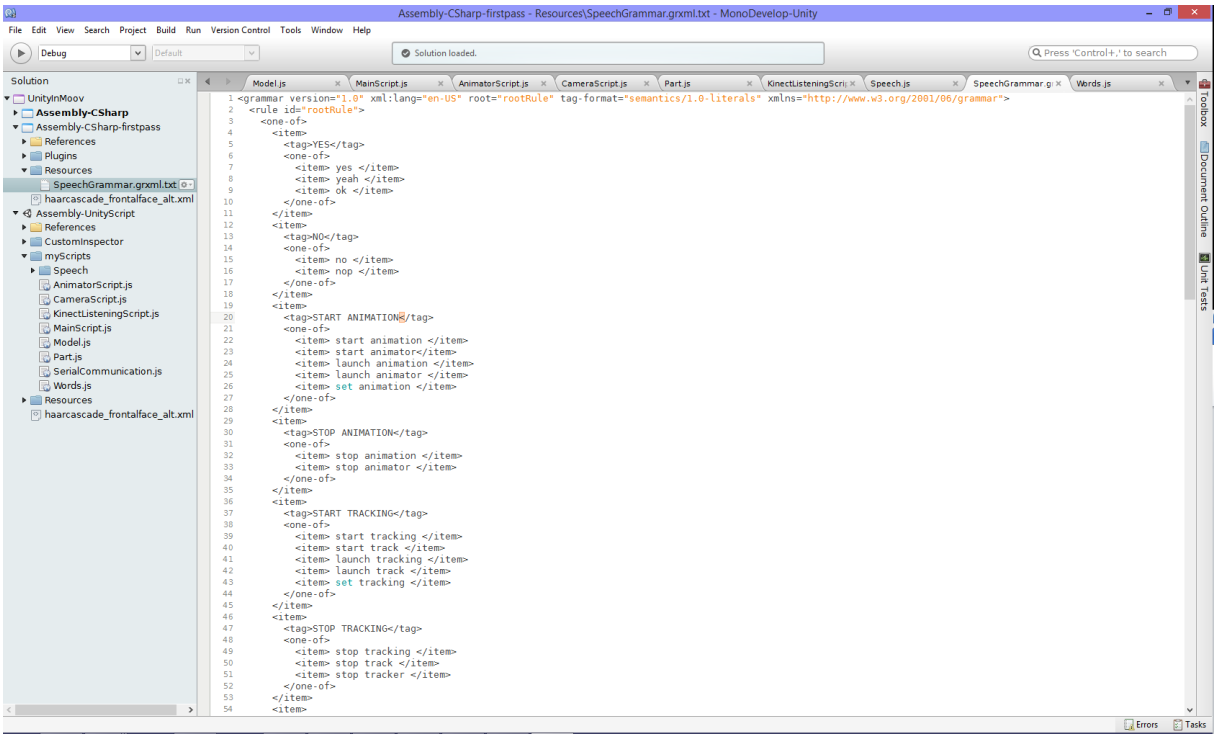
You can add parameters by clicking on the "+" , choose the type of parameter and enter the name.



Here : from Any State, if dance is true, the state Ola will be played.

Now how to add new keywords:

Open Assets->Resources-> SpeechGrammar.grxml



To add a new keywords:

Insert :

```
<item>
    <tag>MY KEYWORD</tag>
    <one-of>
        <item> my key </item>
        <item> my keyword </item>
        <item> keyword </item>
        <item> key </item>
    </one-of>
</item>
```

You can write different words which will be considerate as the same keyword. It increases the probability to detect the keyword.

Then open the Word.js

In the function setPhraseRecognized:

In the switch:

add a case with exactly the same keyword than in the SpeechGrammar file.

inside the case, choose what do you want to do.

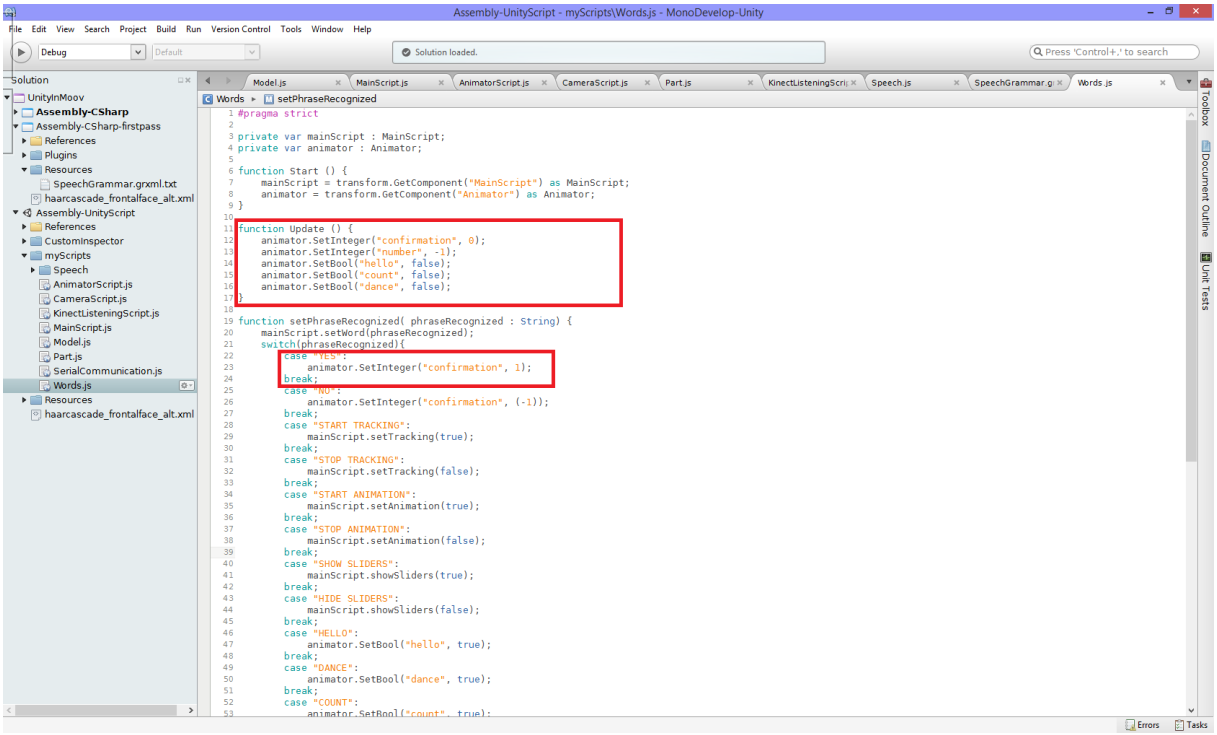
Interact with the interface by calling mainScript function.

Interact with the AnimatorController by calling animator function.

ex: animator.SetBool(“myParameter”, true);

animator.SetInteger(“confirmation”, 1);

With parameters, don’t forget to initialize them each update to their default value.



One last thing, outside of Unity, go in your project folder and delete the SpeechGrammar.grxml from the main folder (not the file in the Assets/Resources). It's for load the new kewords.

That's all!

You should now be able to interact with your robot with yours own keywords.