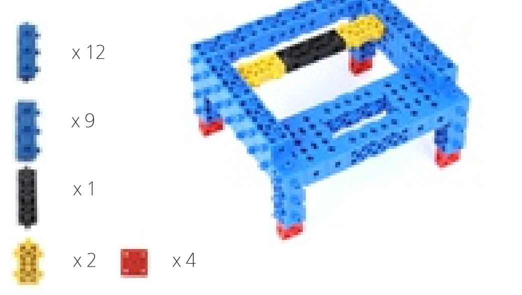
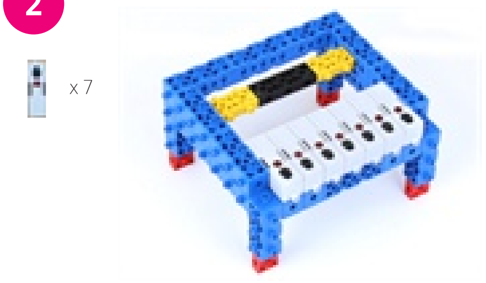
Making a robot

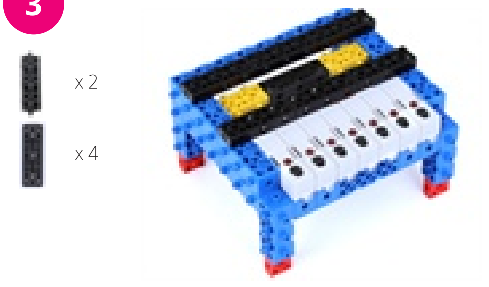
1



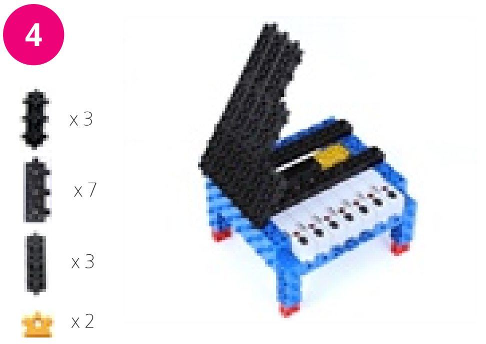
2



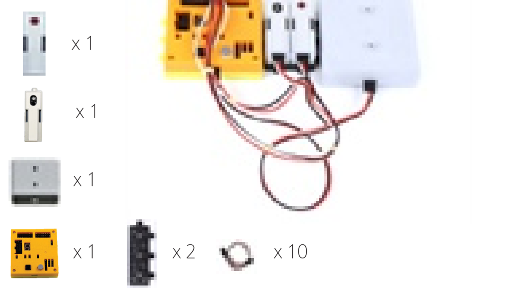
3



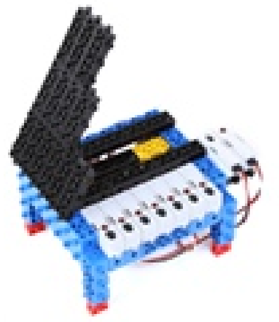
4



5



Complete

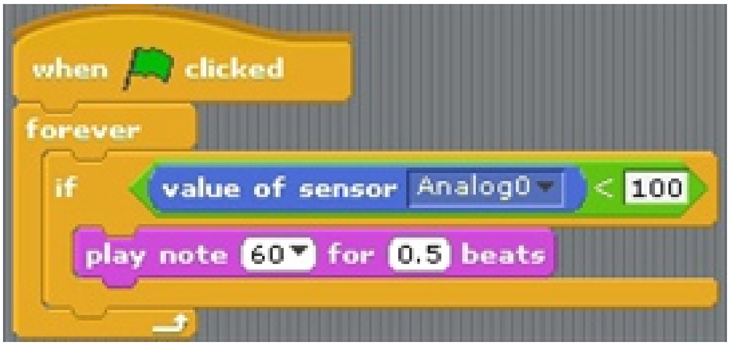


CLASS 4

TORI Piano coding activity

TORI piano sound play

Before making TORI Piano mentioned above, we need to make a program without the recording function that sounds out Do Re Mi Fa So La Ti when the button sensor between 1~7 on the piano is pressed.



```
when ⚑ clicked
forever
    if value of sensor Analog0 < 100
        play note 60 for 0.5 beats
```

TORI Piano playback coding

Let's make a script by dragging in the blocks like the one shown above.
The first block means that the script will be executed when the flag icon is clicked on the scratch screen, and the 'infinite loop' (of 'infinite repeat') block underneath it means that we will continue to check the script in this block indefinitely.

Inside the 'infinite loop' block there is 'If~' block. This block activates the script when certain condition is satisfied. The condition that has to be satisfied in this case is 'value of analog sensor <100'. This means that is the value of the analog sensor is less than 100 (when button sensor is pressed) the script inside will be activated.

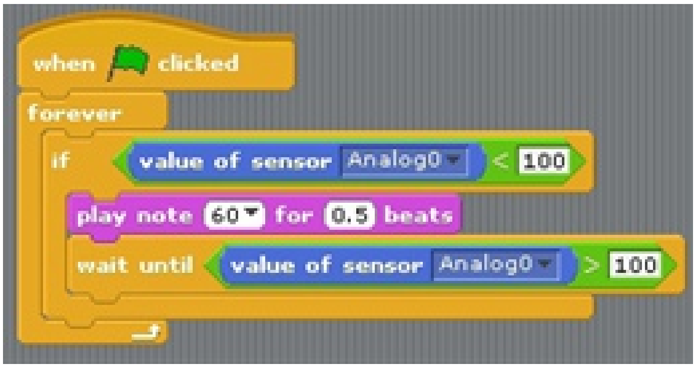
In conclusion, the script above says, "After you press the flag icon on the screen, when the button sensor on analog port 0 is pressed, you will hear the 60th note (warm up) at 0.5 beat," and continue to check the button sensor value it will be.

TORI Piano coding activity

TORI Piano Playback coding

Does the script you created work well?

You will find a strange thing as you are running the preceding script. If you press and hold the button sensor connected to the analog 0 port, you will notice that you are continuing to press and hold the button while you are pressing the button.



```
when ⚑ clicked
forever
    if value of sensor Analog0 < 100
        play note 60 for 0.5 beats
        wait until value of sensor Analog0 > 100
```

Use the script from the previous page to make a script above.

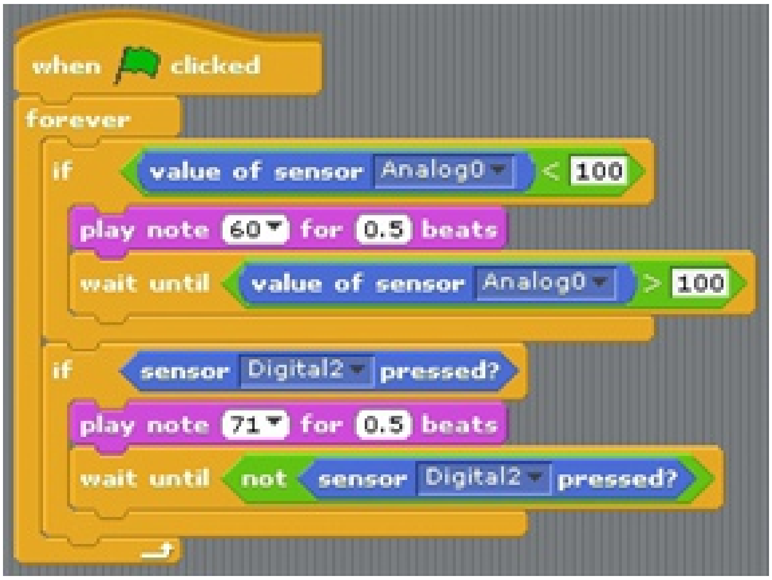
Unlike the previous script, now we have added a "Waiting" block. This block waits for a specific condition, and that condition in this case will be "Analog 0 sensor value > 100".
In other words, if the button sensor on the analogue 0 port is pressed, you will play the 60th note once at 0.5 beats and wait until the value of the analogue 0 sensor is greater than 100 (until you release the button sensor). This will cause the button sensor to be pressed once, wait for the 60th note to sound and then release it, then go up again (because it is an infinite repeat) and press the button sensor. That is, if you hold down the button sensor, the note will not be heard, but only once.Now use the script above to code it so that it will play the sounds from Do to Si.

CLASS 4

TORI Piano coding activity

TORI Piano Playback coding

Does the script you made above function correctly?
If so, we will add another note. It is very similar to the script we made above, so it should be very easy to make.



```
when ⚑ clicked
forever
    if value of sensor Analog0 < 100
        play note 60 for 0.5 beats
        wait until value of sensor Analog0 > 100
    if sensor Digital2 pressed?
        play note 71 for 0.5 beats
        wait until not sensor Digital2 pressed?
```

Create the script above using the script you created on the previous page. You have to add another script that plays the note.

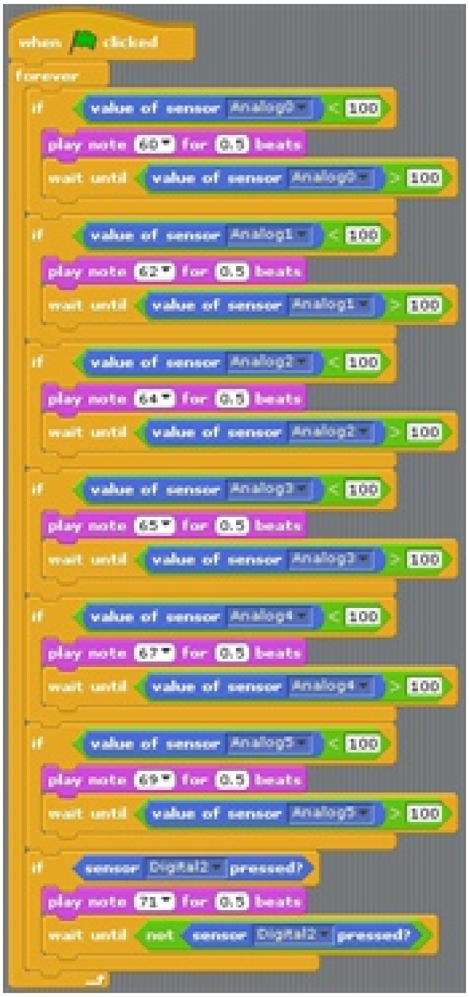
The script to add at this time is a button sensor that outputs 71 notes. Since the main cell has six input analog ports, you need to connect the button sensor to the input digital port 2 to sound 71st note. The ‘If~’ condition for digital 2 sensor is “is the digital sensor pressed?”. Button sensor is ‘true’ in this condition whenever it is pressed. The principle is same for analog port. If digital sensor 2 is pressed, play the 71st note, if not, wait until the hand is released from digital sensor 2. If so, the button sensor connected to input digital port 2 will also play once every time you press and release it.

Now let ‘s make a piano that makes the sounds Do Re Mi Fa So La Ti?

TORI Piano coding activity

TORI Piano Playback coding

Now, let's create a script that connects all the notes between the Do and Si. It will look like the one below.



```
when clicked
forever
  if value of sensor Analog0 < 100
    play note 60 for 0.5 beats
    wait until value of sensor Analog0 > 100
  if value of sensor Analog1 < 100
    play note 62 for 0.5 beats
    wait until value of sensor Analog1 > 100
  if value of sensor Analog2 < 100
    play note 64 for 0.5 beats
    wait until value of sensor Analog2 > 100
  if value of sensor Analog3 < 100
    play note 65 for 0.5 beats
    wait until value of sensor Analog3 > 100
  if value of sensor Analog4 < 100
    play note 67 for 0.5 beats
    wait until value of sensor Analog4 > 100
  if value of sensor Analog5 < 100
    play note 69 for 0.5 beats
    wait until value of sensor Analog5 > 100
  if sensor Digital2 pressed?
    play note 71 for 0.5 beats
    wait until not sensor Digital2 pressed?
```

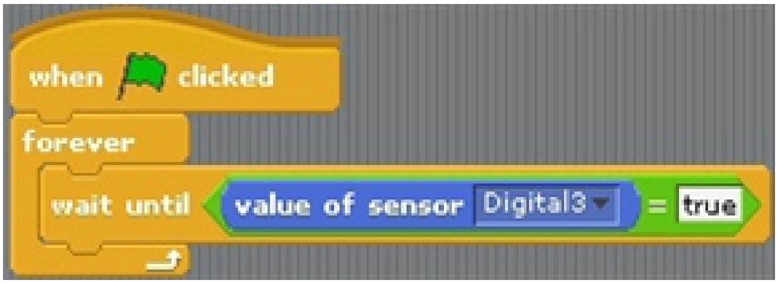
Now shall we start making TORI Piano for both recording and playback?

CLASS 4

TORI Piano coding activity

Activating TORI Piano

First, let’s make a code that is activated when the IR sensor connected to Digital Port 3 is pressed. The early sensor value of Digital Port is “false”. When the user presses the infrared sensor, the sensor value changes to "true." Using this, code “if Digital sensor 3 value = true , then activate the game” .



```
when (flag) clicked
forever
  wait until <(value of sensor Digital3) = true>
```

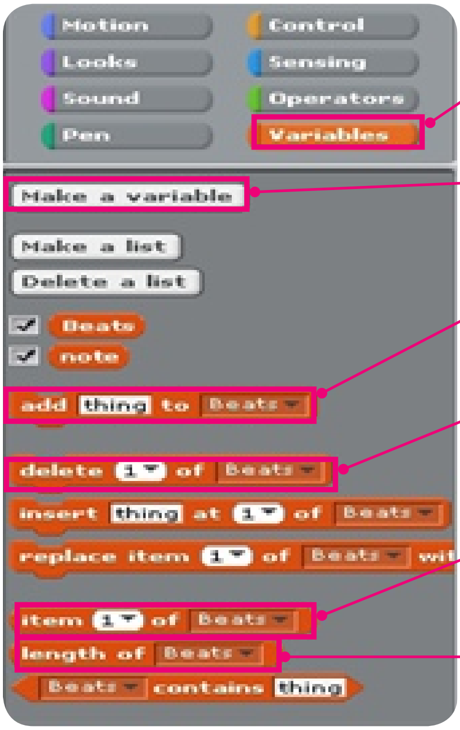
TORI Piano Activation Coding

Drag the blocks in the picture above and make the script.
The first block means that if you click on the flag icon in the scratch screen, the script will be activated. And the ‘infinite loop’ block connected to it below means that it will repeat the script inside this block continuously.
Inside the ‘infinite loop’ block, there is no ‘waiting’ block. This block waits until the certain situation is met. The condition in this case will be “the value of digital sensor 3 = true.” In other words, if the value of the digital 3 sensor becomes true, then when the infrared sensor of digital port 3 is pressed, it exits this block and executes the next command. If so, the above code would say, "if the flag icon is pressed, system will wait for the infrared sensor connected to Digital Port 3 to be pressed."

TORI Piano coding activity

TORI Piano list coding

In order to add the recording function to the TORI piano, we need to use the ‘list.’
List is a function that you can save as much value as you want. The lists you need is the one for the ‘notes’ and another for ‘beats.’ The TORI piano should store the notes of the button sensor you press in the list of 'notes' in order, and store the corresponding notes in the list of 'beats.'



Click the Variables tab to create a list.

Click the Create List button to create a list.

It is a block that adds values to the list.

It is a block that deletes the value of a specific position in the list.

It is a block that represents the value (item) at a specific position in the list.

It is a block that represents the total number of items stored in the list.

TORI Piano List Coding

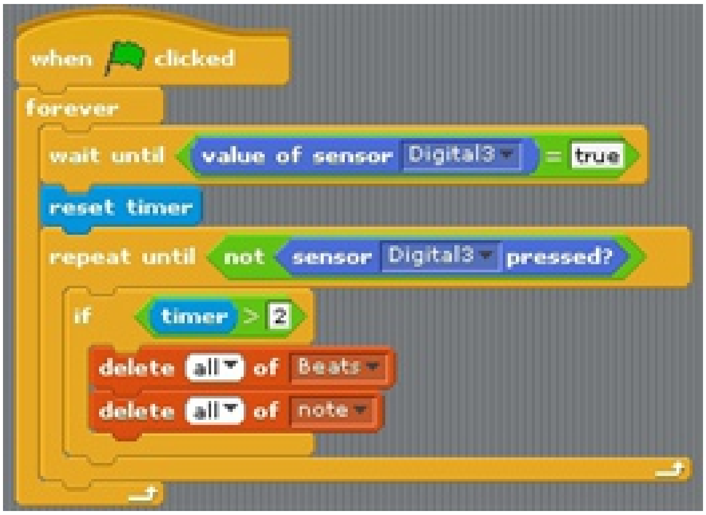
Make a list like the one shown above. ‘Beats’ is responsible for beat and the ‘note’ is the list responsible for the notes. If you made the list like the one above, you have everything you need to move on to the next step. We are going to start utilizing these from the next page!

CLASS 4

TORI Piano coding activity

TORI Piano activation coding

The infrared sensor connected to digital port 3 should be able to play both the role of recording and playback. You can use a timer so that one infrared sensor can do two things. You can think of a timer as a stopwatch in progress in the scratch program. Let's code the program that start recording when the IR sensor of digital port 3 is pressed more than 2 seconds, and plays the recorded sound when pressed less than 2 seconds.



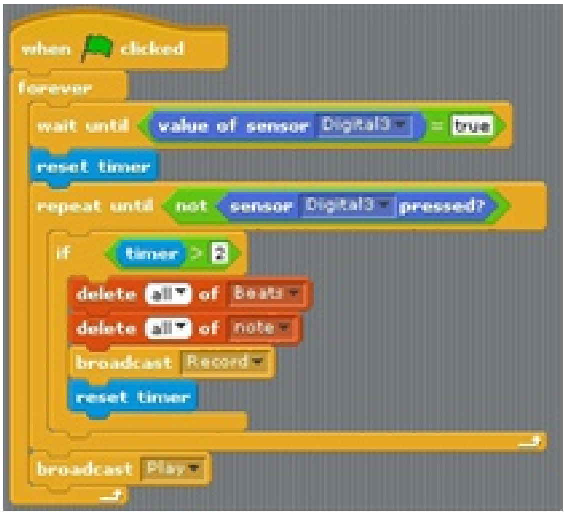
```
when 🏁 clicked
forever
    wait until < (value of sensor Digital3) = true >
    reset timer
    repeat until < not < sensor Digital3 pressed? > >
        if < (timer) > 2 >
            delete all of Beats
            delete all of note
```

Use the script we made previously to make a script like the one above. Reset the timer when the value of the digital 3 sensor is 'true.' Up to this point, you're holding the digital 3 infrared sensor. The 'repeat up to ~' block is the block that repeats the script until the certain condition is met. As you can see above, the condition for this block in this case is "is sensor digital 3 is pressed? Is not." In other words, the block continues to run internal scripts until the user removes the infrared sensor. The 'If~' block inside has the condition of 'time>2.' If timer is greater than 2, the contents saved in 'Beats' and 'note' lists are deleted. This is the process of emptying the list for purpose of recording. In conclusion, if the user presses the digital 3 infrared sensor for more than 2 seconds, the list of 'Beats' and the 'note' list are all deleted.

TORI Piano coding activity

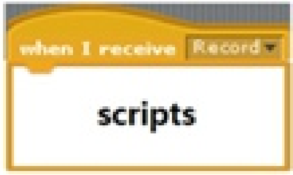
TORI Piano activation coding

If the infrared sensor connected to Digital Port 3 is pressed for more than 2 seconds, it must record, and if you release your finger 2 seconds before, it must send a signal to play. For this, we need to use the 'broadcast' block. 'Broadcast' in the scratch means sending signals. If you send signal, you must be able to receive the signal, right?

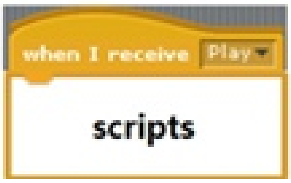


```
when ⚑ clicked
forever
    wait until < value of sensor Digital3 = true >
    reset timer
    repeat until < not < sensor Digital3 pressed? > >
        if < timer > 2 >
            delete all of Beats
            delete all of note
            broadcast Record
            reset timer
    broadcast Play
```

The picture above is the script after the 'broadcast' block is added. If you press Digital 3 sensor for 2 seconds or longer, all the contents stored in the list are deleted, and the Broadcast (Record)' block is executed and the timer is reset. Otherwise (if you do not press for more than 2 seconds) the 'Broadcast (Play)' block is executed.



```
when I receive Record
    scripts
```



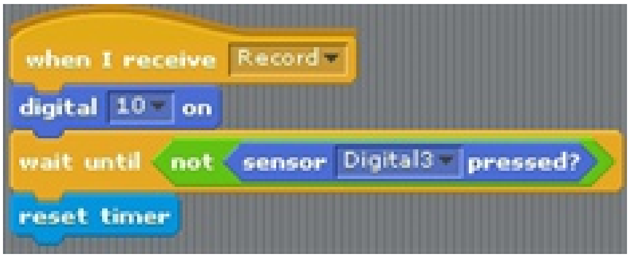
```
when I receive Play
    scripts
```

If 'Record' or "Play' is broadcasted, we need to receive the signal. The picture above is the "receive (Record)" and "receive (Play)" blocks. And below that is the script that will be activated. In other words, according to the time that digital 3 sensor is pressed, it broadcasts 'Record' or 'Play', and the subsequent script is executed depending on which broadcast was received.

CLASS 4

TORI Piano coding activity

TORI piano recording Coding



```
when I receive Record
digital 10 on
wait until not sensor Digital3 pressed?
reset timer
```

Drag the coding locks and make a script like the one above. Did you get it?

First, receive the 'Record' broadcast from the top. Then the bottom blocks are executed in order. The reason I chose 'Record' here is to let you know that it is a script related to recording. When broadcasting, it is helpful to specify the word related to the script to do complicated coding.

Now let's turn on the digital 10 port's output to let you know that recording has begun. You need to press the Digital Port 3 infrared sensor for more than 2 seconds before recording starts in the previous step, so the infrared sensor is still pressed when you receive the 'Record' broadcast. So, wait until the infrared sensor is not pressed (until the input value becomes false). When the user releases the handset, the timer is initialized and recording starts!

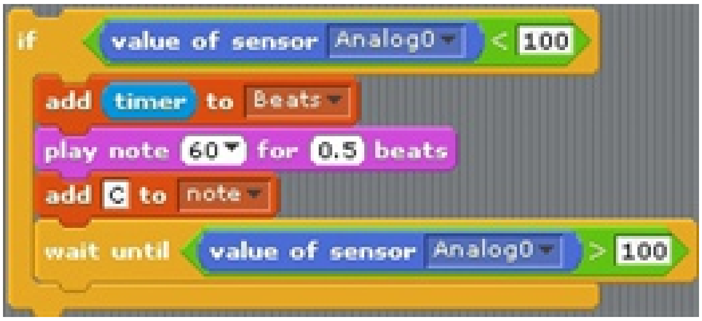
Now the LED on your piano is on and the timer is initialized to start recording. Now on the next page, we will code the script using the list we learned earlier to make a full-scale recording. We will work to make a system that saves the list for the notes by pressing the button sensor before the IR sensor of Digital Port 3 is pressed and turns off LED to announce the end.

First, we will learn about the principle of saving note and beat when the 60th note is played.

TORI Piano coding activity

TORI Piano, repetition of ‘If~’ block

Do you remember the ‘IF~’ block we used previously? It was the block that activates its internal script when certain condition was fulfilled. If so, a block of 'if ~ if' will be created for the notes between Do and Si.
That is, seven ‘IF~” block should be connected. So, what happens to the script inside each 'if~' block? They will be similar to each other. The reason is that if you only need to see the condition of 'Analog Sensor <100', you have to express the actions (save the beat in the 'Beats' list, play the note, and save the note in the 'note' list). So, if you make, "If the value of analogue 0 sensor is <100," block, then the only thing you need to do is to make sure that the script inside the block is accurate.
Then, let’s take a look at which script goes into the “If ‘analog 0 sensor value < 100’.”



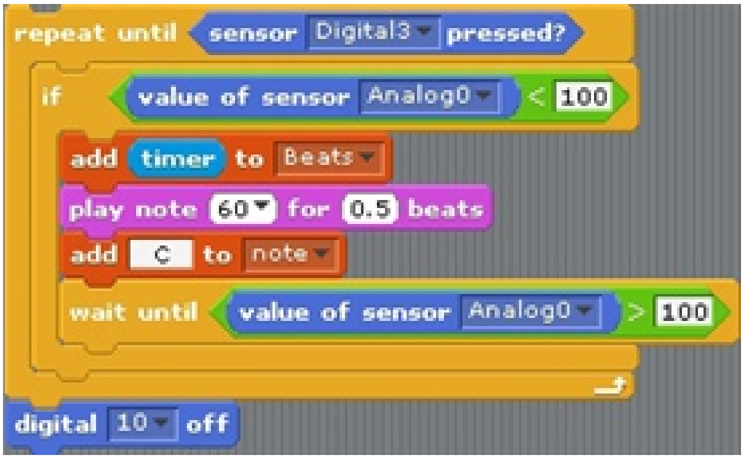
```
if <(value of sensor [Analog0]) < 100>
    add (timer) to [Beats]
    play note 60 for 0.5 beats
    add C to [note]
    wait until <(value of sensor [Analog0]) > 100>
```

Now make a script just like the one above. Do you remember the script that was only able to playback? If you just add the list portion to that list, it will also be able to record! On the previous page, we made a script so that when the recording starts, the timer is reset. In the initialized state, there will be no input and the timer will continue to flow. At this time, if you press the analog 0 button sensor, the timer of that moment will soon be beat. Drag and drop the block that adds this timer value to the ‘Beats’ list. If you press the button sensor again after this, the interval between the timer will be the length of the beat. Then play the 60th note. Then he 60th note is saved in the ‘note’ list. On the list it is saved and displayed as “C.” In the scale, ‘Do’ is shown as ‘C’, so we set the system to save ‘C’ to save the readability. Also, to play the note only once, we add the “wait until ’analog 0 sensor value >100’” block. Then we have a script saying "If you press the analog 0 button switch, the time of the timer is added to the 'Beats' list. After playing the 60 notes, add the note to the 'note' list and wait until the button sensor is released.”

CLASS 4

TORI Piano coding activity

End Tori Piano recording

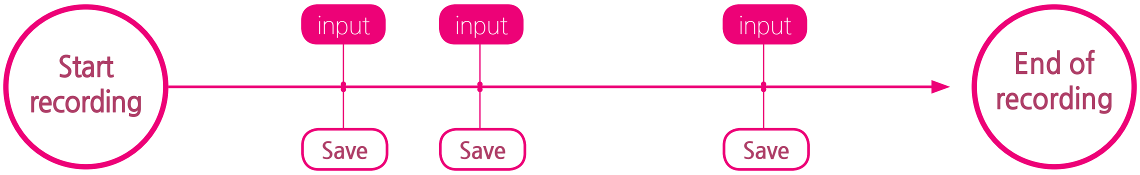


```
repeat until <sensor Digital3 pressed?>
    if <(value of sensor Analog0) < 100>
        add (timer) to Beats
        play note 60 for 0.5 beats
        add C to note
        wait until <(value of sensor Analog0) > 100>
digital 10 off
```

Using the script from previous page, make a script just like the one above. You can record until the Digital Port 3 IR sensor is pressed again. To express this behavior, use the 'Repeat until~" block. This block repeats the internal script until certain condition is met. The condition in this case is 'Is Digital 3 Sensor pressed?"

Which means, it will continue the internal script (recording) until the Digital 3 IR sensor is pressed. After the recording, if you press the digital 3 IR sensor, Digital 10 port output is off. If the recording is done, the LED connected to digital 10 port will be turned off.

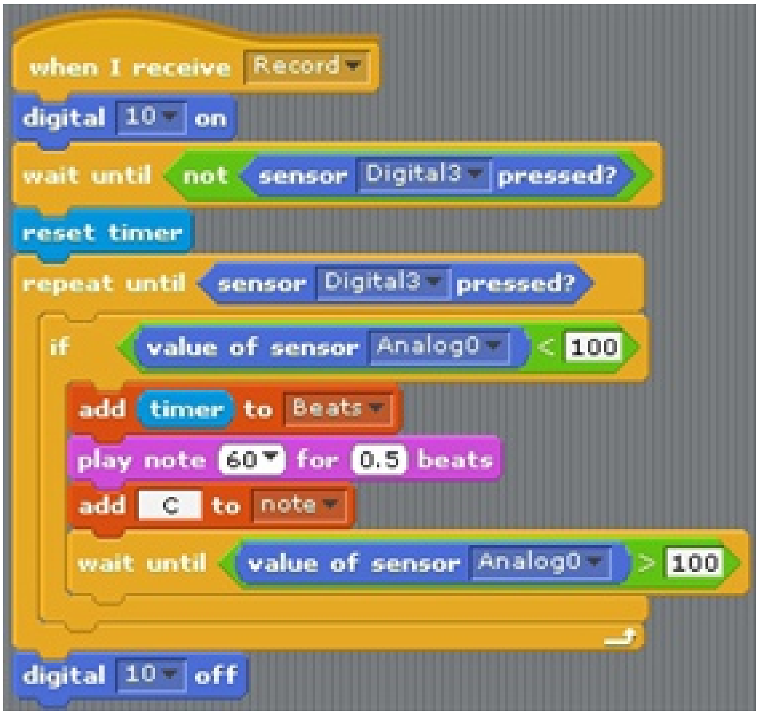
Let me give you an additional explanation on expressing beat with the 'Beats' list. When the recording starts, the timer is initialized and the time continues until recording is finished. In the meantime, if the user presses a note, the note is stored at the point where it was pressed. The idea is that the timer will play from the beginning and at the time the sound is stored.



TORI Piano coding activity

Recording ‘60th’ note

Now you have made an “‘analog 0 senor value <100” script.
Then now let’s assemble the scripts we made so far.



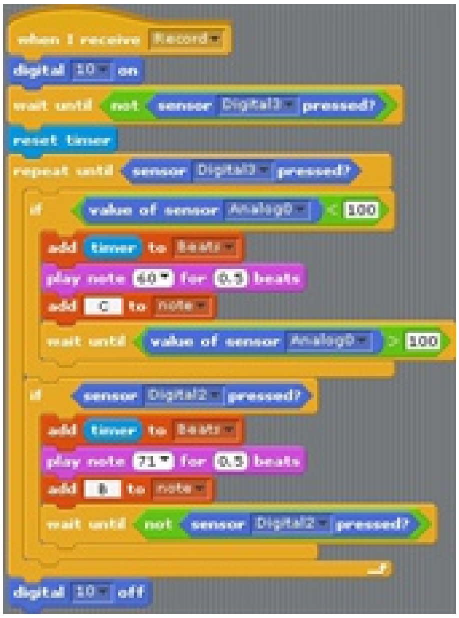
```
when I receive Record
digital 10 on
wait until not sensor Digital3 pressed?
reset timer
repeat until sensor Digital3 pressed?
    if value of sensor Analog0 < 100
        add timer to Beats
        play note 60 for 0.5 beats
        add C to note
        wait until value of sensor Analog0 > 100
digital 10 off
```

The script above is what you should get if you assemble everything. If you start the recording, LED is turned on, and it waits until the use release hand from the digital 3 IR sensor.
After that, the timer resets. The reason for initialization of the timer was explained previously. Add the beats and notes to the list every time the analog 0 button sensor is pressed until the digital 3 infrared sensor is pressed again.

On the next page, we will learn how to code the script when the button sensor value on input digital port 2 Is entered. Analog 0 to analog 5 is very similar to the script above, but the script for playing 71st note is little different, so we will go over that separately.

TORI Piano coding activity

Recording '71st' note



```
when I receive Record
digital 10 on
wait until not sensor Digital3 pressed?
reset timer
repeat until sensor Digital3 pressed?
    if value of sensor Analog0 < 100
        add timer to Beats
        play note 60 for 0.5 beats
        add C to note
        wait until value of sensor Analog0 > 100
    if sensor Digital2 pressed?
        add timer to Beats
        play note 71 for 0.5 beats
        add B to note
        wait until not sensor Digital2 pressed?
digital 10 off
```

Using a script from previous page, make a script shown above.

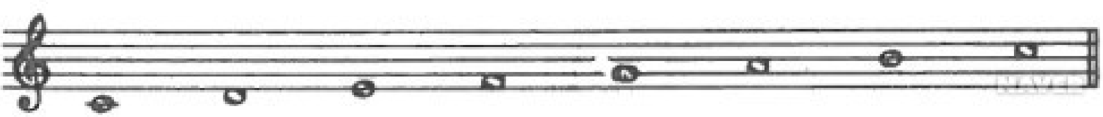
The contents of the input of the digital 2-button sensor were covered in the previous simple playback piano. The principle is the same as the button sensor connected to the analog port. The condition for 'If~' button for digital 2 button sensor is 'is digital 2 sensor pressed?' When you press the Digital 2 sensor, the timer time of that press is stored in the 'Beats' list and it plays 71st note. Then it adds 71st note to the 'notes' list. With the same reason as the 60th note, 71st note is saved as 'B' into the list. Wait until the digital 2 button sensor value becomes 'false' because you should play it only once.

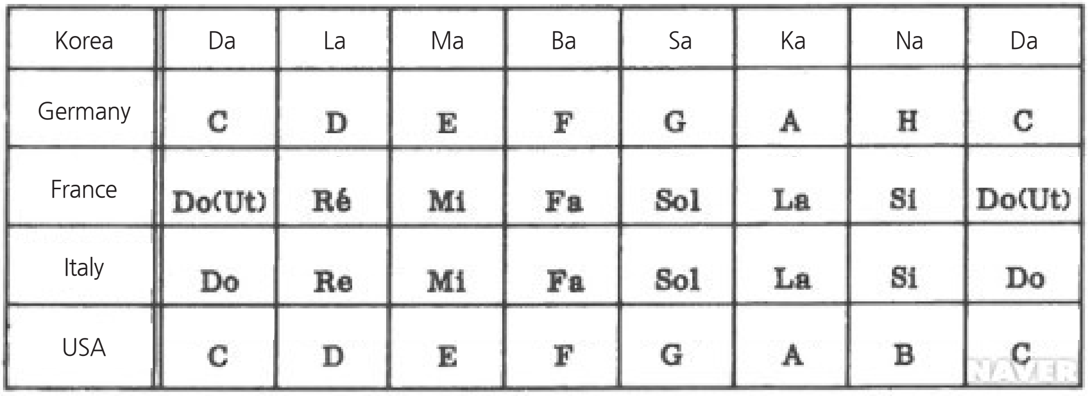
The difference between an analog sensor and a digital sensor is that the analog sensor determines the size of the value and the digital sensor determines if the value is true or false, principle and contents are the same.

TORI Piano coding activity

Learn sonic name

Do you remember that we set ‘Do’ as “C’ and ‘Si’ as ‘B’?
On this page we will briefly learn the sonic names of each note.





Korea	Da	La	Ma	Ba	Sa	Ka	Na	Da
Germany	C	D	E	F	G	A	H	C
France	Do(Ut)	Ré	Mi	Fa	Sol	La	Si	Do(Ut)
Italy	Do	Re	Mi	Fa	Sol	La	Si	Do
USA	C	D	E	F	G	A	B	C

Source: Classical Music Term Dictionary

The sonic name is about the pitch of the sound. It’s a fixed name for each note. It’s a different concept than the notes that depends on the first note. When you're coding a piano, you'll be storing notes in the 'note' list. At this time, please set the value to take into consideration the sonic name

Then, on the next page, create a script that records the entire sound between C and B. Applying what you've learned in the past will help you!

If you can’t remember what to do, please refer back to the previous pages.

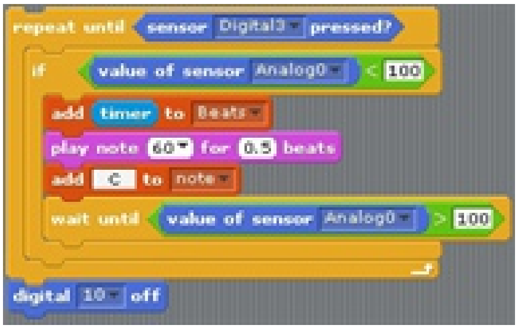
CLASS 4

TORI Piano coding activity

Code a script for recording C~B

Using the script on the right, code a script that will record from C to B.

Make sure that the port of the analog sensor and the port of the digital sensor are changed, and check that the note names stored in the 'note' list are set considering the sound.

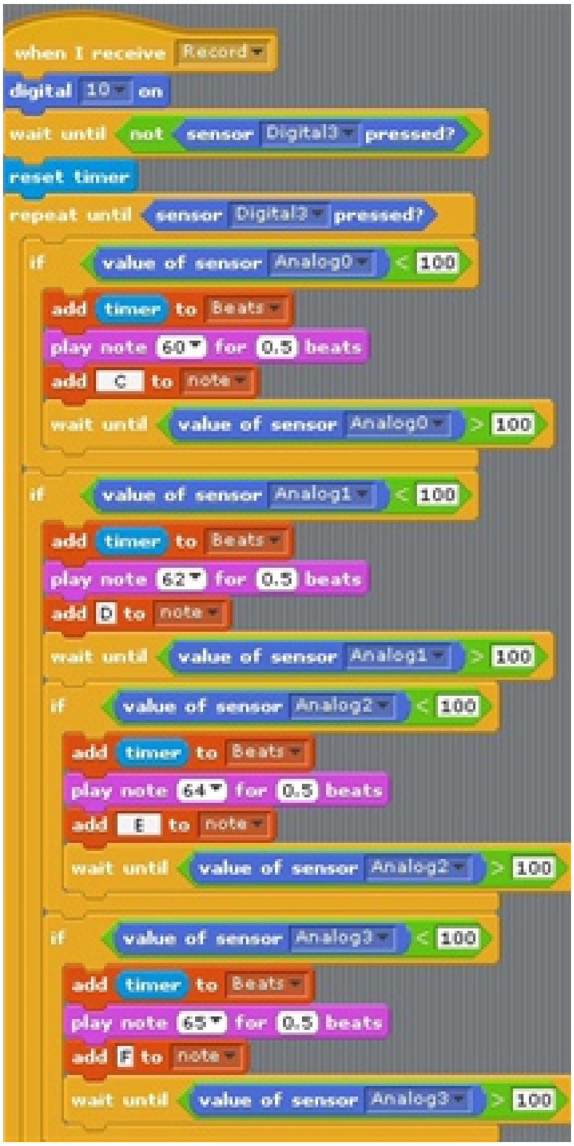


```
repeat until sensor Digital3 pressed?
    if value of sensor Analog0 < 100
        add timer to Beats
        play note 60 for 0.5 beats
        add C to note
        wait until value of sensor Analog0 > 100
digital 10 off
```

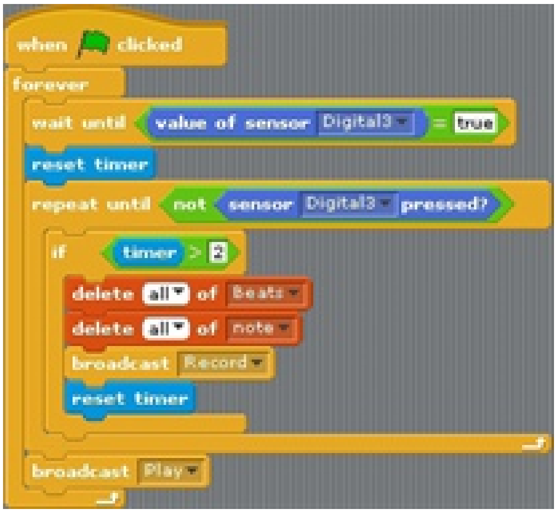
TORI Piano coding activity

TORI Piano active activation and recording coding.

The script for the recording part of the TORI Piano is expressed as below



```
when I receive Record
digital 10 on
wait until not sensor Digital3 pressed?
reset timer
repeat until sensor Digital3 pressed?
  if value of sensor Analog0 < 100
    add timer to Beats
    play note 60 for 0.5 beats
    add C to note
    wait until value of sensor Analog0 > 100
  if value of sensor Analog1 < 100
    add timer to Beats
    play note 62 for 0.5 beats
    add D to note
    wait until value of sensor Analog1 > 100
    if value of sensor Analog2 < 100
      add timer to Beats
      play note 64 for 0.5 beats
      add E to note
      wait until value of sensor Analog2 > 100
    if value of sensor Analog3 < 100
      add timer to Beats
      play note 65 for 0.5 beats
      add F to note
      wait until value of sensor Analog3 > 100
```



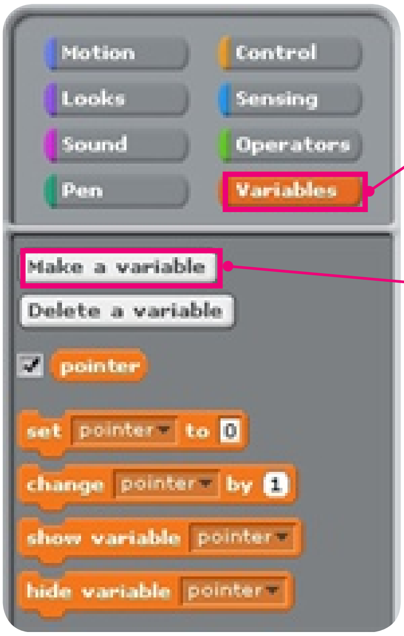
```
when ⚑ clicked
forever
  wait until value of sensor Digital3 = true
  reset timer
  repeat until not sensor Digital3 pressed?
    if timer > 2
      delete all of Beats
      delete all of note
      broadcast Record
      reset timer
  broadcast Play
```

CLASS 4

TORI Piano coding activity

TORI Piano variable coding

Now, let's create a script for the part that plays Tori Piano. Pre-recorded beats and notes are stored in the 'Beats' and 'note' lists. You need a role to 'point and play' the notes and notes stored in each list in order to create a TORI piano play script. Here we have to use the variable we learned earlier, the 'pointer' variable. Variables can easily be thought of as storage space.



Click the Variables tab to create a Variables

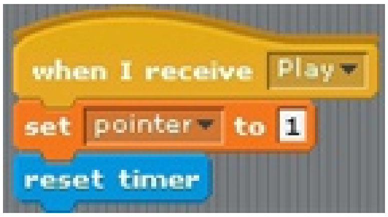
Click the Create Variables button to create a Variables.

TORI Piano variable coding

Make the variable just like the picture above. Here, ‘pointer’ is a variable that represents the value saved at ’note’, and ‘Beats’ lists. We will uses the variable we made here in the next page.

TORI Piano coding activity

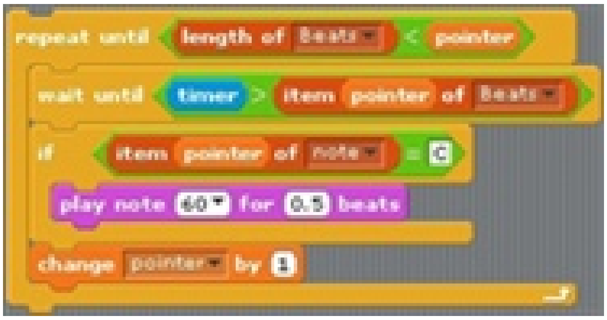
TORI Piano Playback coding



```
when I receive Play
set pointer to 1
reset timer
```

Go ahead and make a script like the one above. Did you get it?

First, on the top you receive a 'Play' signal. Then the blocks below that will be activated in order. The reason why we saved it as 'Play' in this case is also to show that it is related to recording. Save 1 at 'pointer' variable. This is to show that it is the first item in the 'note' and 'Beats' list. Then reset the timer since it needs to read the pre-recorded timer from the very beginning.



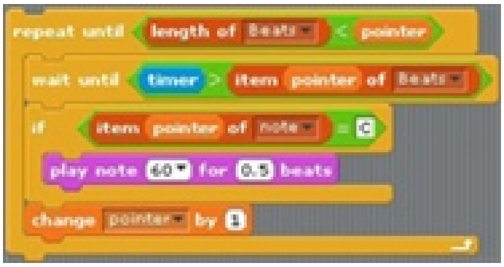
```
repeat until length of Beats < pointer
    wait until timer > item pointer of Beats
    if item pointer of note = C
        play note 60 for 0.5 beats
    change pointer by 1
```

Should we start making the script above? The script to play the recorded file is not so complicated. The playback script will point to an item in the 'Beats' list where the timer is stored with the 'pointer' variable during the timer's time. At first, it points to the value of the first item. Then play the corresponding to the item 1 in the 'note; list. Then add 1 to the 'pointer' variable. The above script is when the item stored in the 'note' list is 'C.' Use the 'repeat to~' block for now. The condition for this block will be 'Beats size < 'pointer'.' The reason for using this condition is to let 'pointer' point to the total number of items stored in the 'Beats' list. If the value of 'pointer' exceeds the number of 'Beats' list items, it exits the block until 'repeat to ~' and ends the playback script.

CLASS 4

TORI Piano coding activity

TORI Piano playback coding



```
repeat until < (length of [Beats]) < (pointer) >
    wait until < (timer) > (item (pointer) of [Beats]) >
    if < (item (pointer) of [note]) = [C] >
        play note (60) for (0.5) beats
    change [pointer] by (1)
```

Let's continue with the explanation for the script on the previous page.
Should we take a look into the script inside the 'repeat to~' block? The timer is initialized and continues to flow when it is played. And wait until the timer is larger than the item value of 'Beats' pointed to by the value stored in 'Pointer' (initially 1). The moment it gets bigger, it gets out of the 'waiting' block. After that, if the item value in 'note' list that is pointed by 'pointer' is 'C', you will play 60th notes.

Do you understand? Let me give you an example to help you understand. Assume that the files recorded in the pre-recorded file include 'C' in the item at position 1 in the 'note' list, '3.000' in the item at position 1 in the 'Beats' list, and the total number of items in both lists is 10. Try it. When playback starts, the first 'pointer' is set to 1, and the timer is initialized and continues to flow.

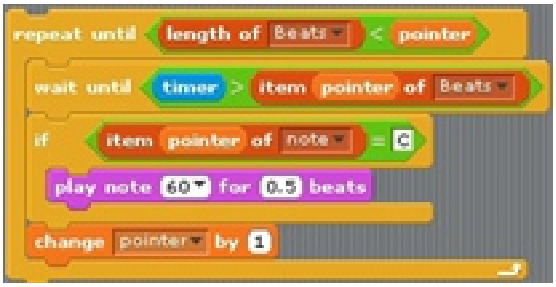
Now let's go to 'repeat to ~' block. Repeat the inner script until the total number of item in 'Beats' is less than the value stored in 'pointer' variable. Currently, the total number of item in 'Beats' is 10, and the 'pointer' value is 1, which satisfies the condition. Now you are running an internal script. The "Wait until 'Timer> Beats 'pointer' position item'" block means that the timer that is running now is waiting for the current timer to be larger than the item in the 'pointer' position in the 'Beats' list. Currently, the value of 'pointer' is 1, which means that the timer waits until it is larger than the item at position 1 in the 'Beats' list (3.000 seconds). As the timer goes down, the moment you pass '3.000' seconds, you exit the 'Waiting' block. And compare it to the condition of the if block. If it satisfies the condition, it plays 60th note.

※ The whole script can be found at http://www.robotori.com/web_eng -> Moretips -> Manual ->EDU ->Logic boost CODING CLASS 4 ->download whole script

TORI Piano coding activity

Coding playback scripts for C~B

Use the script on the right to code the playback script until C~B. There is nothing to change except to add a loop of 'if ~' blocks. You can create a total of 7 blocks from C to B by connecting the condition of 'if' to 'block'. Make it!



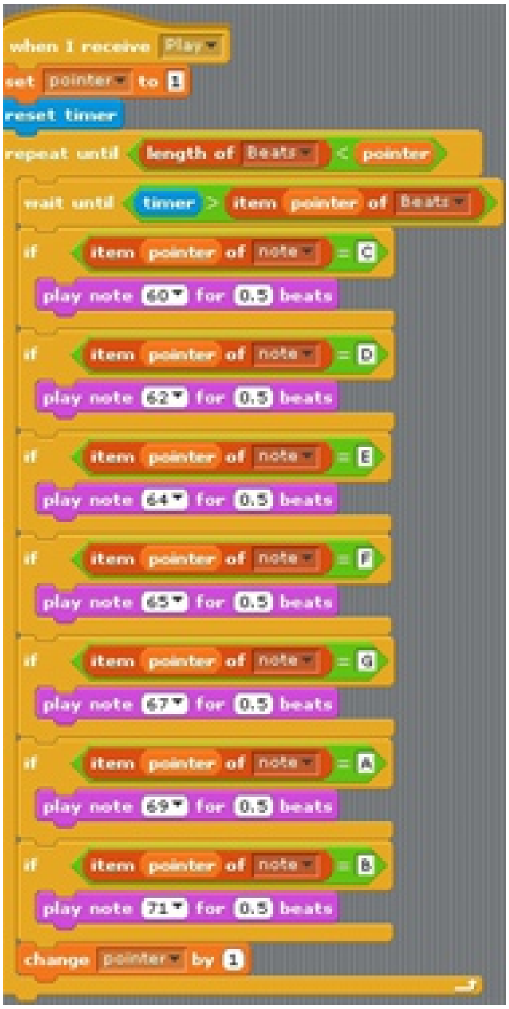
```
repeat until  length of Beats  <  pointer
  wait until  timer  >  item  pointer  of Beats
  if  item  pointer  of note  =  C
    play note 60 for 0.5 beats
  change pointer by 1
```

CLASS 4

TORI Piano coding activity

TORI Piano play coding

The playback script for TORI Piano will look like the figure below!



```
when I receive Play
set pointer to 1
reset timer
repeat until < length of Beats < pointer >
    wait until < timer > item pointer of Beats >
    if < item pointer of note = C >
        play note 60 for 0.5 beats
    if < item pointer of note = D >
        play note 62 for 0.5 beats
    if < item pointer of note = E >
        play note 64 for 0.5 beats
    if < item pointer of note = F >
        play note 65 for 0.5 beats
    if < item pointer of note = G >
        play note 67 for 0.5 beats
    if < item pointer of note = A >
        play note 69 for 0.5 beats
    if < item pointer of note = B >
        play note 71 for 0.5 beats
    change pointer by 1
```

The whole script can be found at http://www.robotori.com/web_eng -> Moretips -> Manual ->EDU ->Logic boost CODING CLASS 4 ->download whole script

TORI Piano coding activity

TORI Piano Play

Now run the coded TORI Piano program to see if it works. If it does not work, please check the beginning of the book to see if there are any errors. Let's play your own with recording and playback functions.
Let’s play your own piece with recording and playback functions.

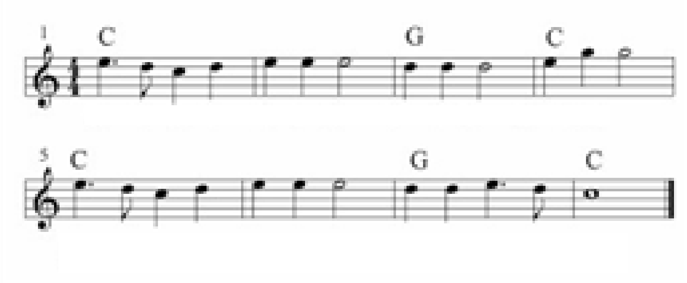
CLASS 4

TORI Piano Problem 1

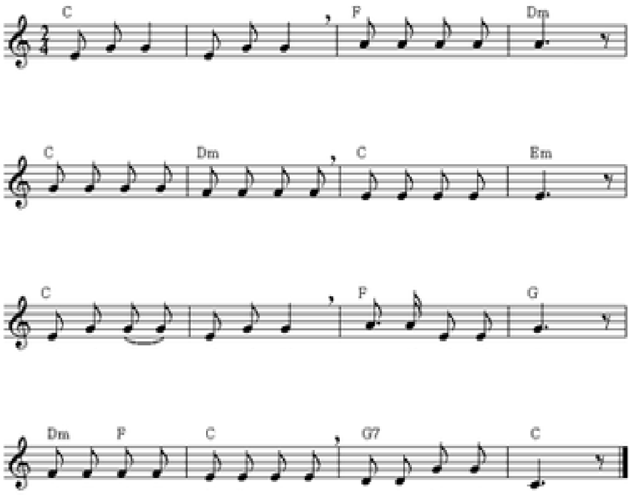
TORI Piano Problem 1

Have you tried playing TORI Piano using the recording and playback functions? Now, let's try recording the same score while watching the score below.

The music below is ‘Airplane.’ You can play this with TORI Piano. Go ahead and try to play it.



TORI Piano Problem 2



TORI Piano Problem 3

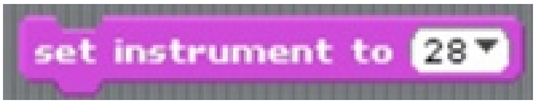
TORI Piano Problem 3

Were you able to play the songs in the previous page? This time, try to use the TORI piano to hear not only the piano but also the sounds of other instruments. On the Sound tab, find the desired block of instruments and combine it with the original script. Let's try it.

Commentary

TORI Piano problem 3 commentary

Locate the block that sets up the instrument in the sound tab and try to connect it to the bottom of the "Record '" block. Find and drag the block shown below.



```
set instrument to 28▾
```

The block above is currently set to use the 28th instrument.
You can select various instruments by pressing the black arrow next to the number. You should set up another instrument only when recording. Because, when you play it, it reproduces the recorded material as it is. Find the instrument you want and set it to sound!

The whole script can be found at http://www.robotori.com/web_eng -> Moretips -> Manual ->EDU ->Logic boost CODING CLASS 4 ->download whole script

Introduction

Artificial intelligence and autonomous vehicles.

You have often heard the term 'artificial intelligence' in society today. Artificial intelligence is entering our lives at a very fast pace, telling what a computer can think and judge like a person. Artificial intelligence can not only do most of the things that people do, but it also gives people ideas that they cannot think of and can do things they cannot do.

One of the applications of this artificial intelligence is 'autonomous driving car'. I'll go into more details about autonomous vehicles later. You can now learn about the basic principles of artificial intelligence by creating autonomous vehicles. Let's see how artificial intelligence is used by creating autonomous vehicles

Now let's go ahead and make an autonomous vehicle!

SCRATCH CODING KIT

Logic Boost

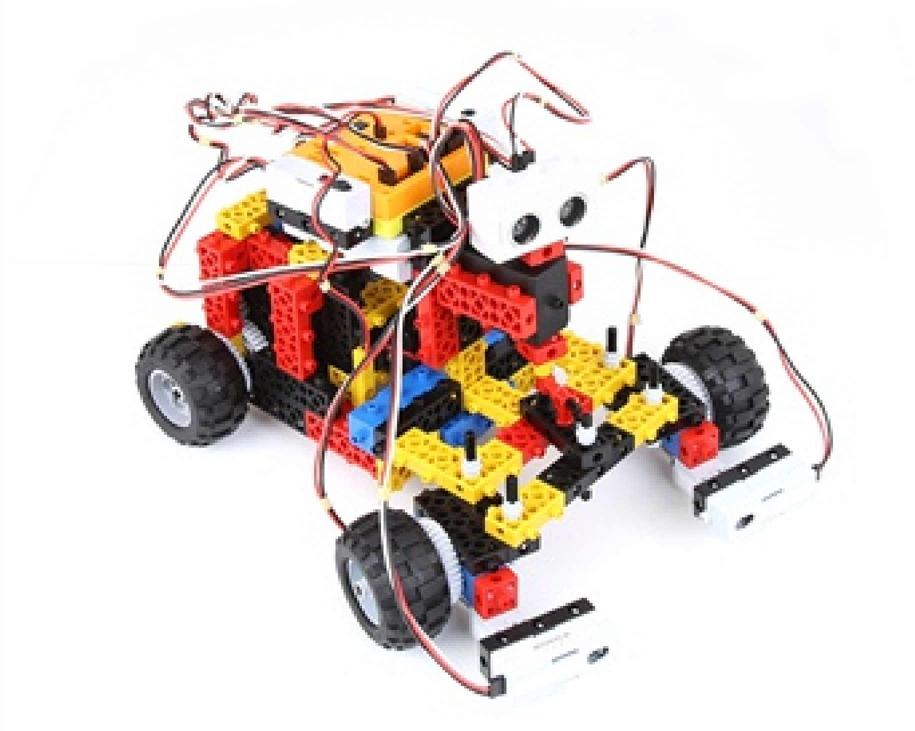
TORI Autonomous Vehicle

LESSON 7

introduction

Tori Autonomous Vehicle

Let's create a self-driving car using various sensors! Let's create a robot that recognizes obstacles using an infrared sensor and an ultrasonic sensor and judges itself to go in the direction without obstacles.



There is a 'differential gear unit' and a 'steering wheel' that allow the Tori autonomous vehicle to move like a real car. First, let's look at these two things and let them learn what role they play by creating a robot yourself.

Let's see how we can move the car and control the direction.

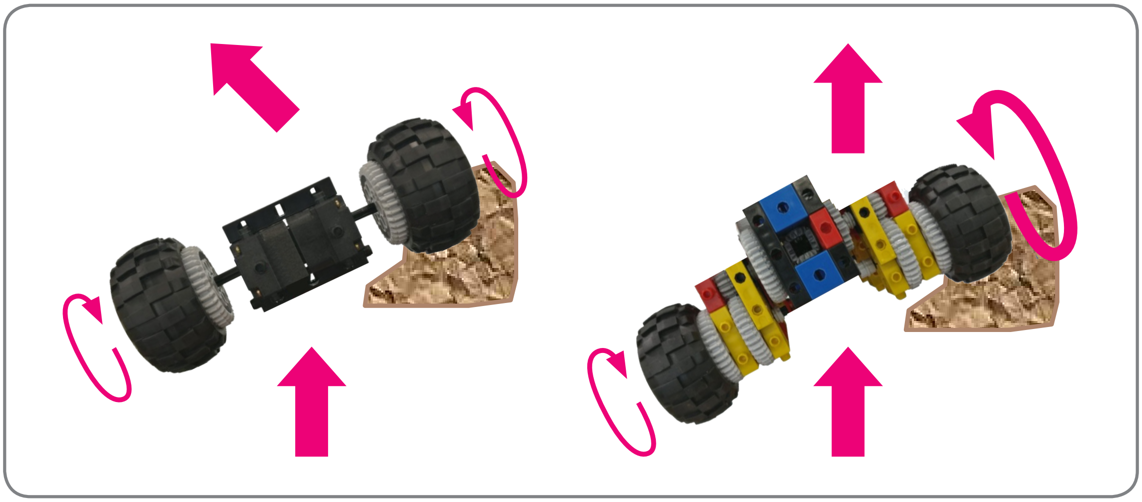
Concept

Differential gear unit

A differential gear unit is a device that is used to transmit moving forces in real cars. So far you've created robots that use two wheels to move around. All the robots that use two wheels use two DC motors. I put the wheels on each DC motor one by one, and I made the robot to operate two wheels.

However, with a differential gear unit, not only can one DC motor deliver the power of movement to both wheels, but also the power delivered to each wheel can be different! Therefore, even when the floor is uneven, the two wheels can work differently and work smoothly.

When two DC motors are used, the power to turn the wheels per DC motor is fixed by a constant force, so it is possible to move forward in a flat place, but in a rugged position, it cannot move forward.



If you use two DC motors as shown in the picture above, the two wheels always rotate with the same force, so you cannot go straight to the rugged place.

However, using a differential gear system gives the wheels on the side with the obstacles more power so the car can go straight! These differential gears are essential for making autonomous vehicles and can be a basic factor.

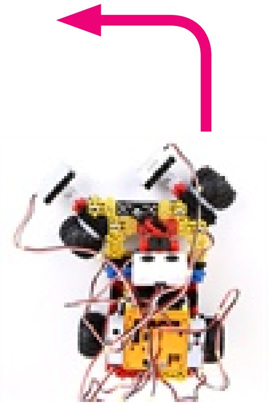
Concept

Steering Wheel

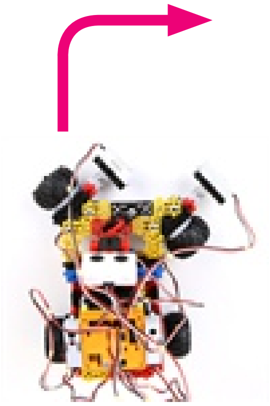
Do you understand the differential gear you learned earlier? Now let's see what is called a steering wheel that is necessary for an autonomous vehicle to operate.

The driver must turn the handle to change the direction of the car to the left or right. The handle of this car is called the steering wheel. This steering wheel can be used to control direction.

In the TORI autonomous vehicle, I will use the servo motor to express the steering wheel behavior. I will connect the servo motor and the front wheel so that I can code the car to run left or right depending on the angle of the servo motor.



Steering wheel when going left



Steering wheel when going right

In the picture above, turn the steering wheel to the left, turn on the DC motor and the car will go to the left. Turn the steering wheel to the right, turn on the DC motor and the car will go to the right. With the steering wheel, you can change the direction of your car when it runs.

The steering wheel is the direction key of the car. This is because you can control the direction of the car depending on how you control the direction keys.

CLASS 4

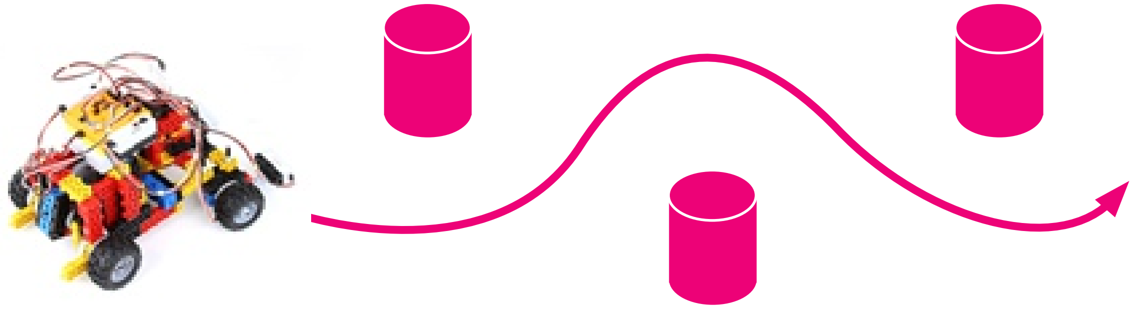
Concept

Autonomous Vehicle

So far, we have learned about the 'differential gearing' and the 'steering wheel' necessary for the car to work. Both of these factors were driving the car. Now let's see what it takes to have a car named autonomous.

An autonomous vehicle is a car that automatically detects the condition of the road without the driver controlling the brakes, handles, and accelerator pedals. The car itself works by recognizing the surrounding situation, judging it according to the situation and minimizing the operation of the driver.

You can learn about 'situation recognition and judgment' and 'motion by judgment' that are used in real autonomous vehicles. From now on we will use infrared sensors and ultrasonic sensors to detect obstacles, find roads without obstacles and code the car to move on its own.



Please concentrate on how well your self-driven trolley car recognizes the situation, how to judge it, how to judge it, and how to operate it.

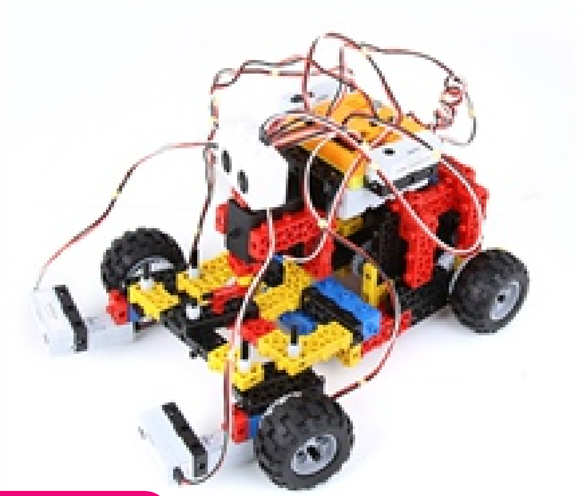
Let's try to think about autonomous driving by creating and operating an autonomous driving car which is one of the important elements in future society!

Now let's go on to make the Tori aoutonomous vehicle!

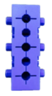
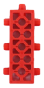
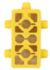
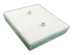
Activities

Tori Autonomous Vehicle

Create your self-judging and moving tori autonomous vehicle



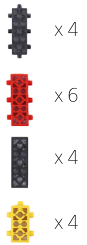
Autonomous Vehicle Material list

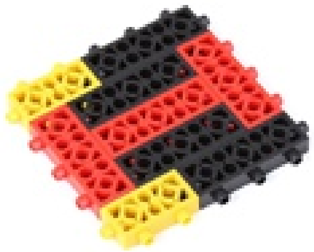
												
Diamond H6 x 4	Diamond V8 x 2	Diamond V6 x 1	Rubi 8 x 5	Rubi 7 x 8	Rubi 4 x 8	Rubi 6 x 13	Rubi 2 x 3	Rubi 0 x 15	Mini 2 x 22	Link x 4	Mini 1 x 8	Servo x 2
Bevel x 4	Sawtooth 24 x 2	Sawtooth 12 x 2	Sawtooth 36 x 7	Short connector x 6	Middle connectorx 18	Vertical connector x2	Motor connector x 1	A 14 x 3	A 23 x 4	A 45 x 9	A 64 x 4	Mainboard 128 x 1
												
Batterey x 1	DC motor x 1	Soung Sensor x 1	IR SENSOR x 4	Button Sensor x 1	ULTRASONIC SENSOR x 1	4pin connectorx 1	differential gear 36 x 1	Servo motor x 1	3pin connectorx 8	Off-road wheel x 4		

Activities

Making Tori Autonomous Vehicle

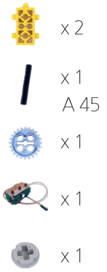
1

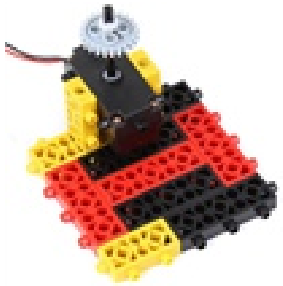




Make two of the same model

2

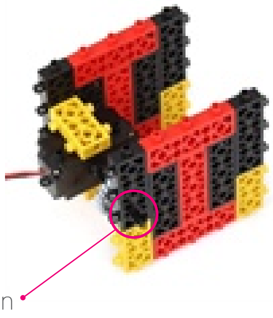




Assemble with model 1

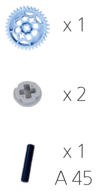
3

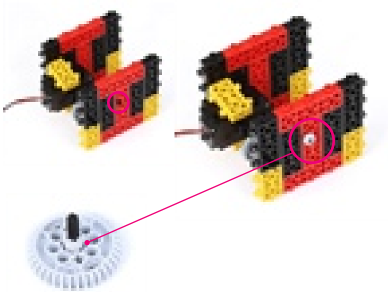
Assemble model 1 and 2 like the one in the picture



caution

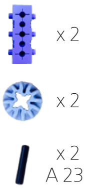
4

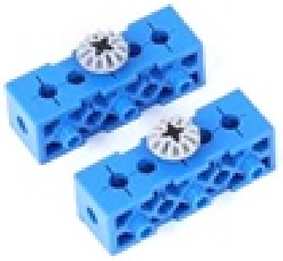




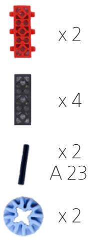
5

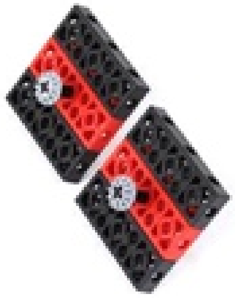
5-1





5-2



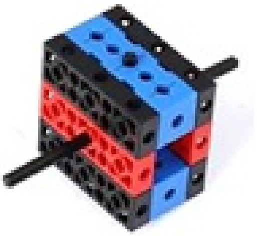


Activities

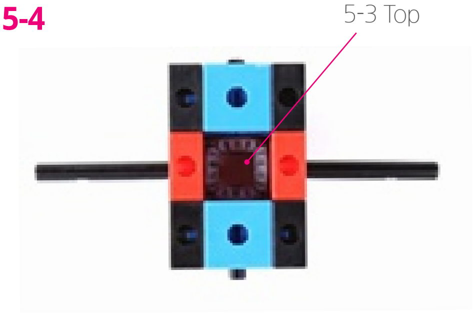
Tori Autonomous Vehicle

5 **5-3**

Assemble 5-1 model and 5-2 model like it is shown in the picture



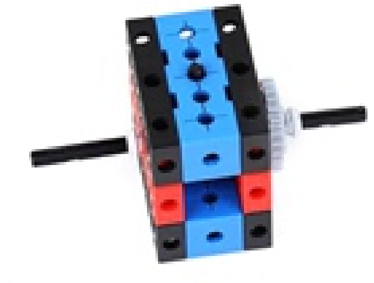
5-4



5-5

x 1

x 2



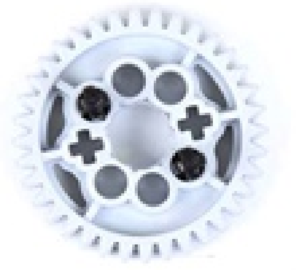
5-6

x 1

x 1

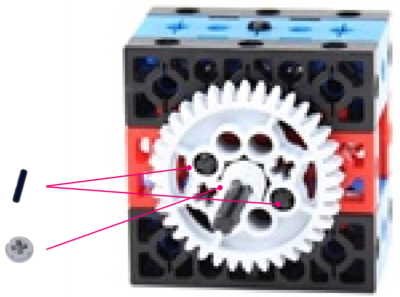
x 2
A 14

differential gear 36



5-7

Fix it to the position so it cannot move.

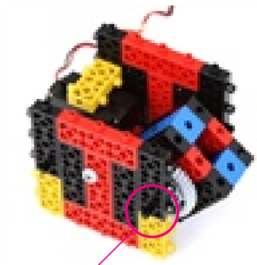


Activities

Making Tori Autonomous Vehicle

6

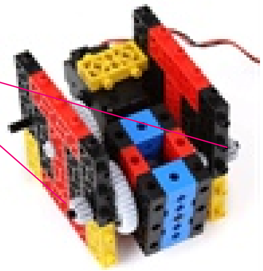
Assemble model 4 and 5-8 like it is shown In the figure.



Assembly location

7

x 2



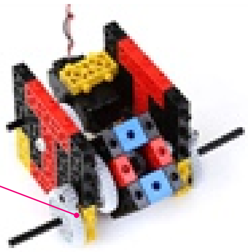
8

8-1

x 2

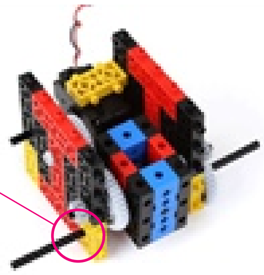
x 2

x 2
A 64



8-2

Sawtooth 36 Assembly location



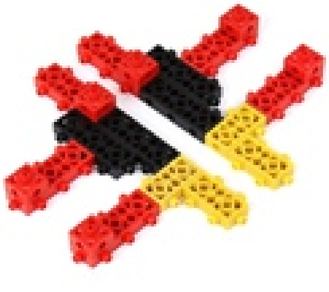
9

x 6

x 4

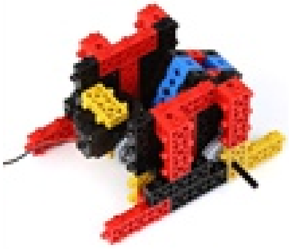
x 4

x 6



10

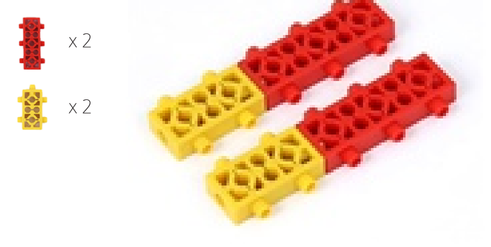
Assemble model 8and 9 like it is shown in the picture.



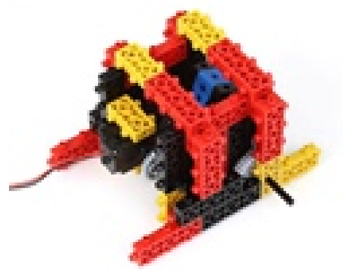
Activities

Tori Autonomous Vehicle

11

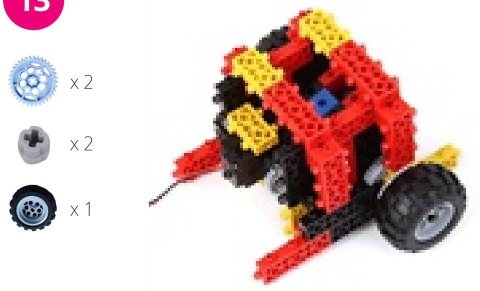


12

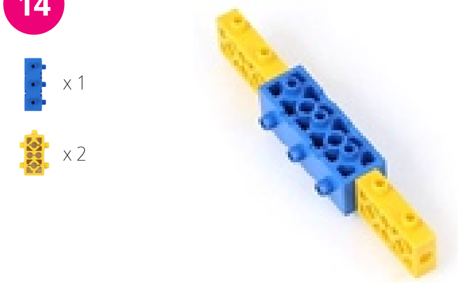


Assemble model 10 and
11 like it is shown in the picture

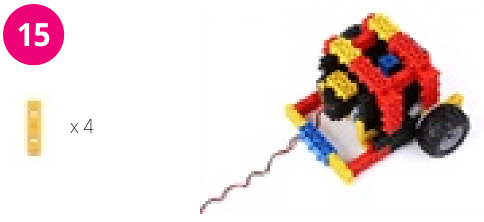
13



14

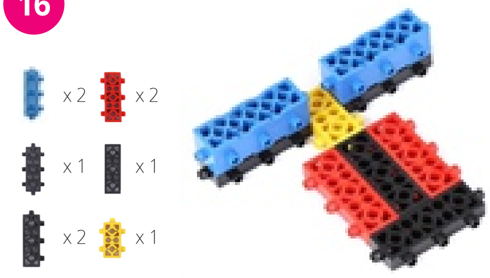


15



Assemble model134 and
13 like it is shown in the figure.
Do the same thing for the back using the
connection jack.

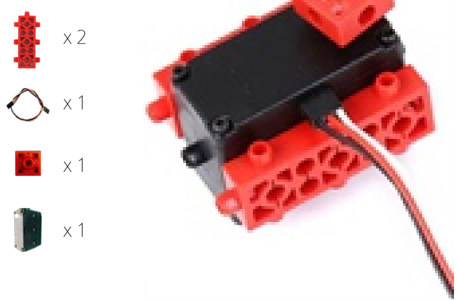
16



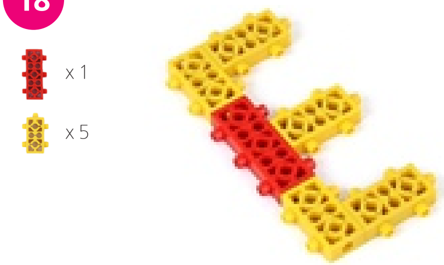
Activities

Making Tori autonomous vehicle

17

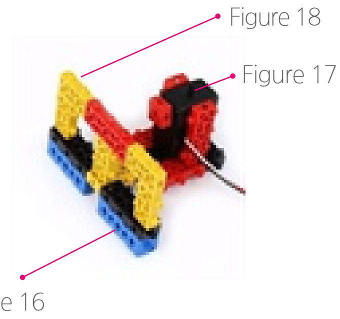


18

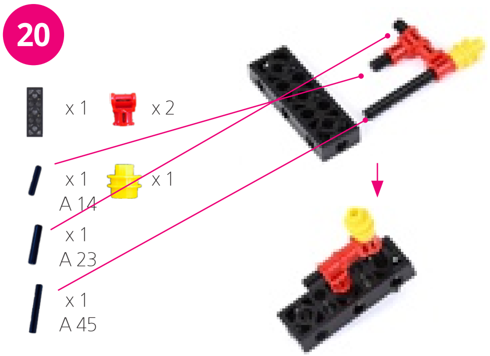


19

Assemble model 16, 17 and 18 like it is shown In the figure.

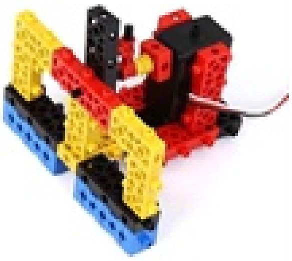


20

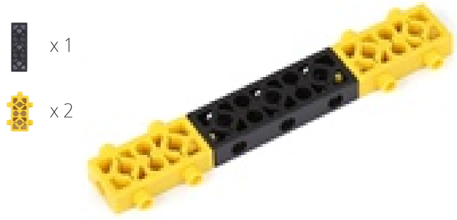


21

Assemble model 19 and 20 like it is shown In the figure.



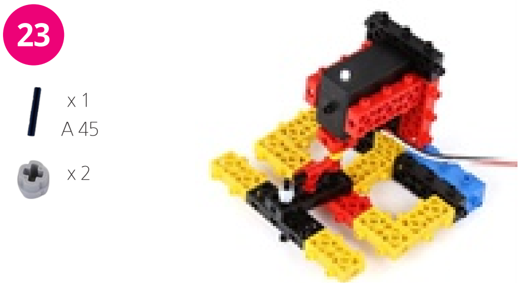
22



Activities

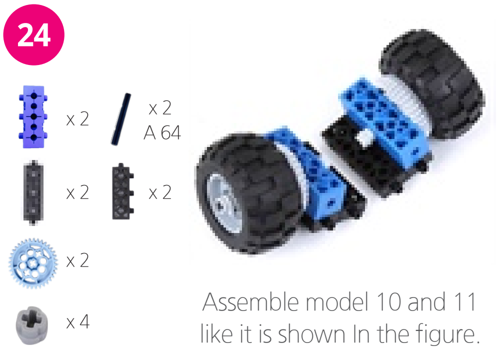
Making Tori autonomous vehicle

23



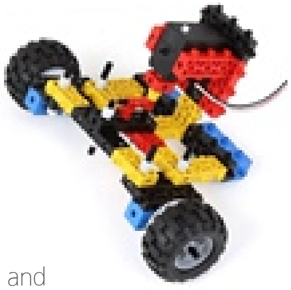
Assemble model 21 and
22 like it is shown In the figure.

24



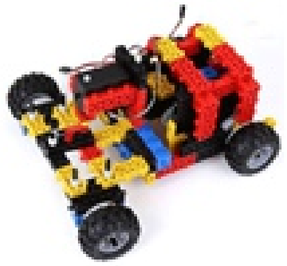
Assemble model 10 and 11
like it is shown In the figure.

25



Assemble model 22 and
24 like it is shown In the figure.

26



Assemble model 15 and
25 like it is shown In the figure.

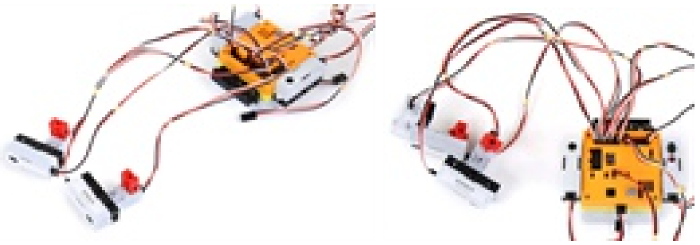
27

Left floor detection IR sensor Analog 0
left front detection IR sensor analog 2
Right floor detection IR sensor Analog 1

Right front detection IR sensor Analog 3
Ultrasonic sensor analog 5
Button sensor Digital 3

SONG GENERATOR sensor Digital 10
Servo motor output terminal end port

IR Sensor*4
ULTRASONIC SENSOR *1
button sensor *1
SONG GENERATOR *1

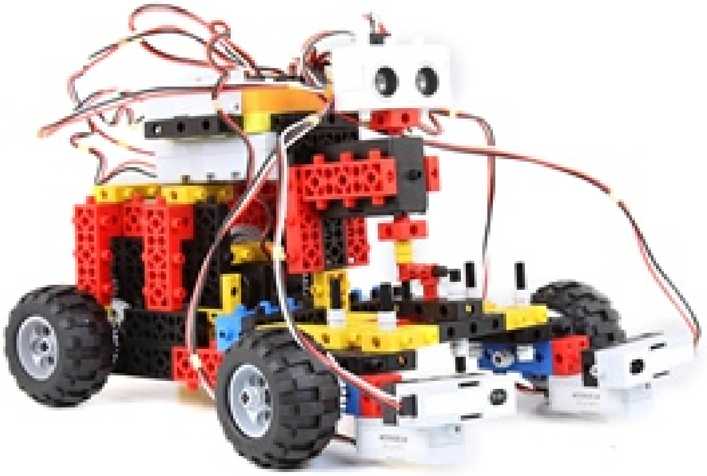


CLASS 4

Activities

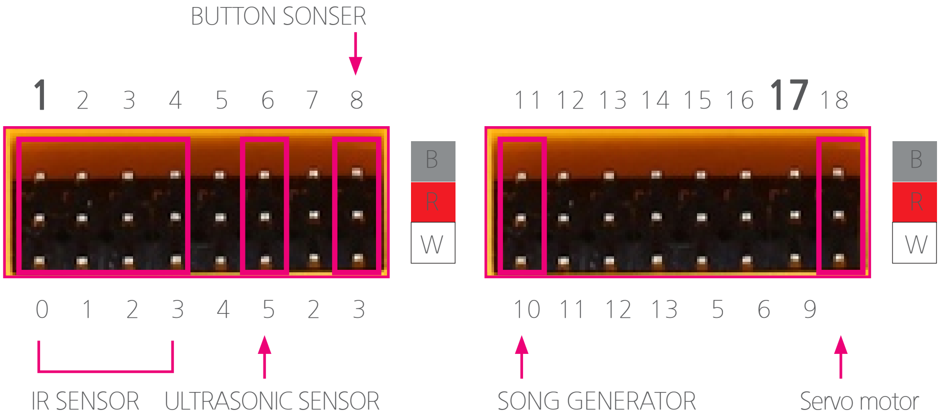
Making Tori Autonomous Vehicle

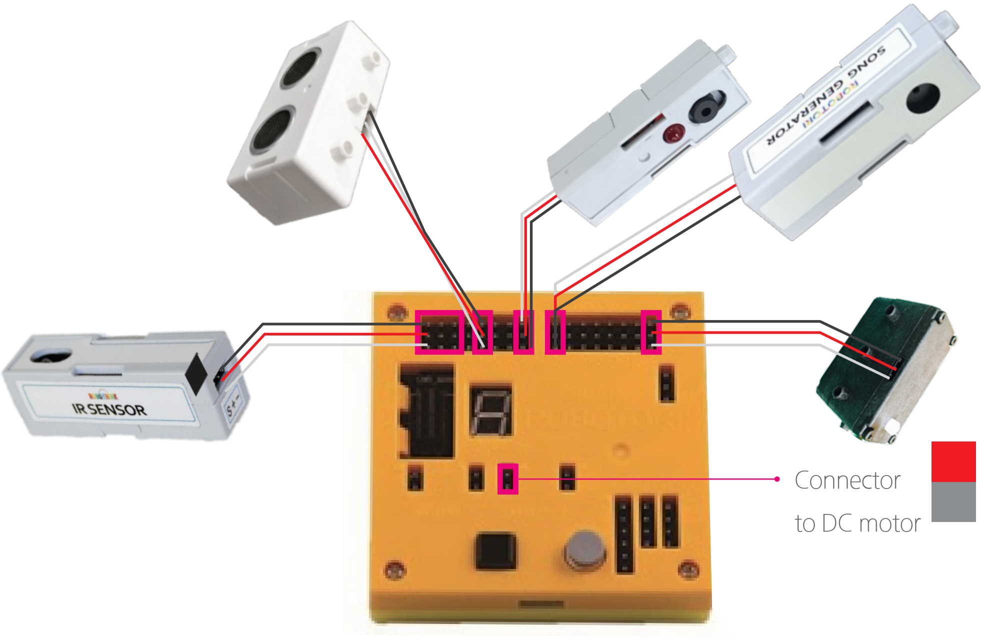
Complete



Connecting Main cell

Connecting Main cell

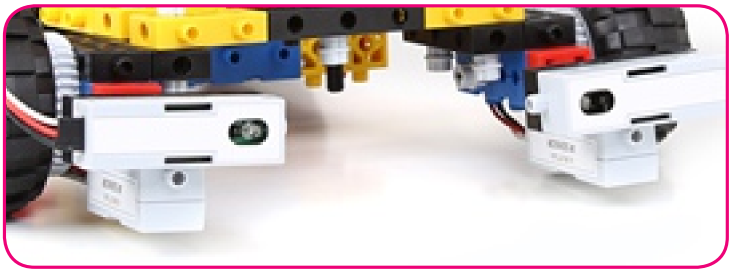




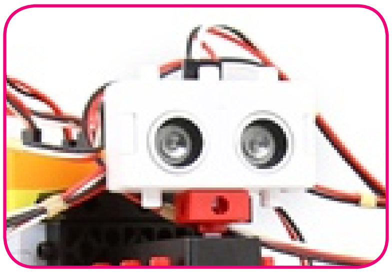
Activities

Coding Tori Autonomous Vehicle

Now, let's try to code a real self-driving car. Your Tori autonomous vehicle is a car robot that uses four infrared sensors and one ultrasonic sensor. Once you are aware of the situation with each input sensor, you can then use the differential gear to move the vehicle to an unobstructed location.



Location with IR sensor



Location with ULTRASONIC sensor

The Tori autonomous vehicle uses two infrared sensors to detect the floor. These infrared sensors are used to make sure the car does not fall off by checking whether the floor is up or down.

Two other infrared sensors are installed in front of the car. These two infrared sensors are used to detect obstacles and determine the location of obstacles.

And, using an ultrasonic sensor, it can detect obstacles and judge their position. Ultrasonic sensors allow it to more precisely locate obstacles.

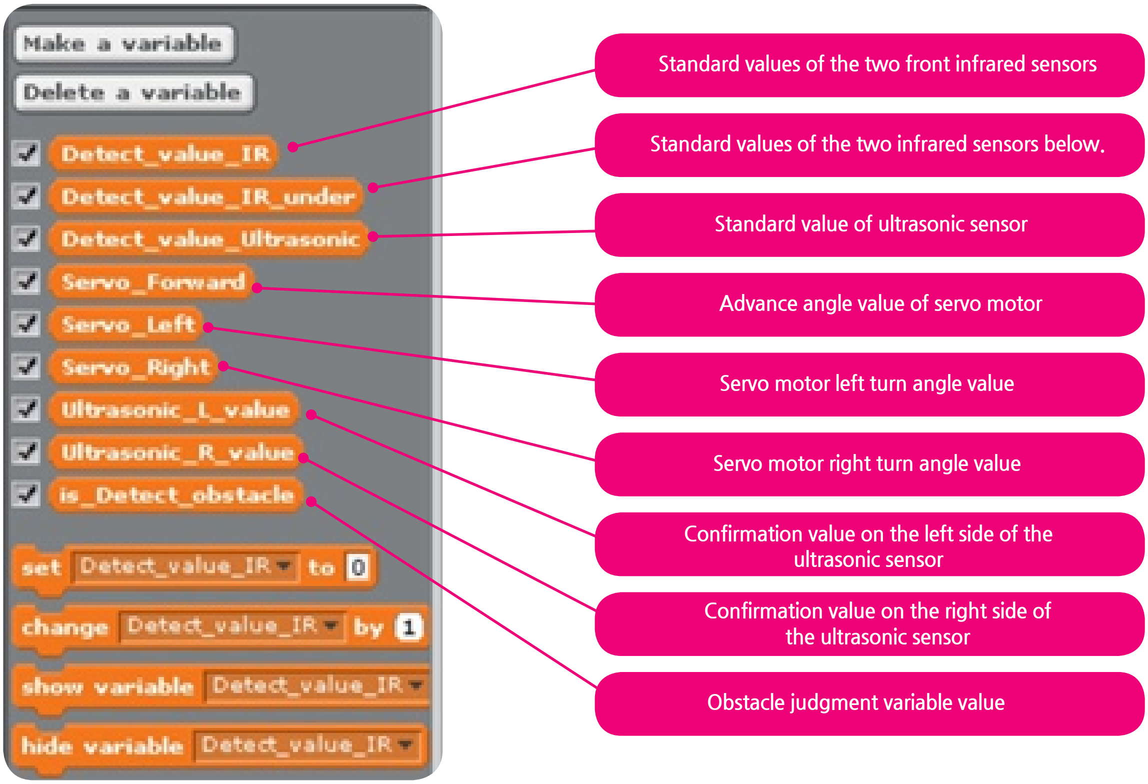
We'll learn about the role each sensor plays and learn how to code it. Take advantage of the concepts you've learned so far.

First, Let's learn about the variables we need to code Tori autonomous vehicle.

Activities

Setting Tori Autonomous Vehicle Variables.

Now let's set up the variables we need to do the coding once. You need a fairly large number of variables to code autonomous vehicles. Find out what variables you need and learn what each one does. First, go to the Variables tab and make the following variables!



Make a variable

Delete a variable

☑ Detect_value_IR

☑ Detect_value_IR_under

☑ Detect_value_Ultrasonic

☑ Servo_Forward

☑ Servo_Left

☑ Servo_Right

☑ Ultrasonic_L_value

☑ Ultrasonic_R_value

☑ is_Detect_obstacle

set Detect_value_IR ▾ to 0

change Detect_value_IR ▾ by 1

show variable Detect_value_IR ▾

hide variable Detect_value_IR ▾

Standard values of the two front infrared sensors

Standard values of the two infrared sensors below.

Standard value of ultrasonic sensor

Advance angle value of servo motor

Servo motor left turn angle value

Servo motor right turn angle value

Confirmation value on the left side of the ultrasonic sensor

Confirmation value on the right side of the ultrasonic sensor

Obstacle judgment variable value

Have you tried creating the variable shown above? Variable names have been given meaning by their respective roles. If the names of the above variables are too difficult, you can change them to any name you like. However, the role of the variable should not change.

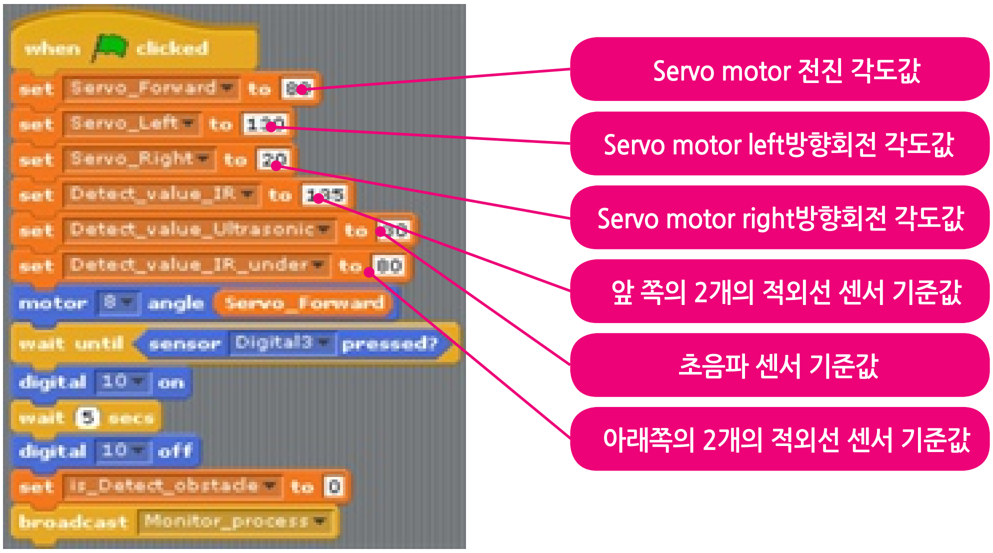
We made the parameters for the infrared sensor, the ultrasonic sensor and the servo motor and the necessary parameters to judge the obstacle. Now try coding the first values of these variables! Be sure to understand the meaning of each variable and set the appropriate initial value!

Activities

Setting the first value of the variable

Now you have to set the initial values of each variable. When you press the Run (activation) button, you need to set the angle of the servo motor and the reference value of the sensor first. Let's set the value according to the role of each variable!

Then, after pressing the Run button, press the button sensor connected to the tori autonomous vehicle, the Song Generator will be able to broadcast a script to announce the autonomous driving start and to start the situation recognition and judgment. Create a script like the one below.



```
when clicked
set Servo_Forward to 80
set Servo_Left to 130
set Servo_Right to 20
set Detect_value_IR to 135
set Detect_value_Ultrasonic to 30
set Detect_value_IR_under to 80
motor 8 angle Servo_Forward
wait until sensor Digital3 pressed?
digital 10 on
wait 5 secs
digital 10 off
set is_Detect_obstacle to 0
broadcast Monitor_process
```

Servo motor 전진 각도값
Servo motor left방향회전 각도값
Servo motor right방향회전 각도값
앞 쪽의 2개의 적외선 센서 기준값
초음파 센서 기준값
아래쪽의 2개의 적외선 센서 기준값

The initial value of each variable depends on the sensors and servo motors you have, so you have to change them manually. Please refer to the explanation above for setting the initial value of each variable. When you finish setting the variable, first control the angle to move the servo motor forward. After that. Wait until button sensor is pressed. Now when the button sensor is depressed, the SONG GENERATOR will make a five second sound and announce the start of the car.

The 'is_Detect_obstacle' recognizes the obstacle and sets '0' as the initial value for the necessary variables to judge. If you broadcast a script called 'motor_process' that starts the situation recognition and judgment, the first setting is finished. Now create a script for 'Monitor_process'!

Activities

Use the IR sensor to judge after recognizing the situation

Now you are going to code how to operate using the values entered by the infrared and ultrasonic sensors of the Tori autonomous vehicle. First, create a script about how the infrared sensor recognizes the situation. This will be the script for the last moniotr_process in the previous script, right?

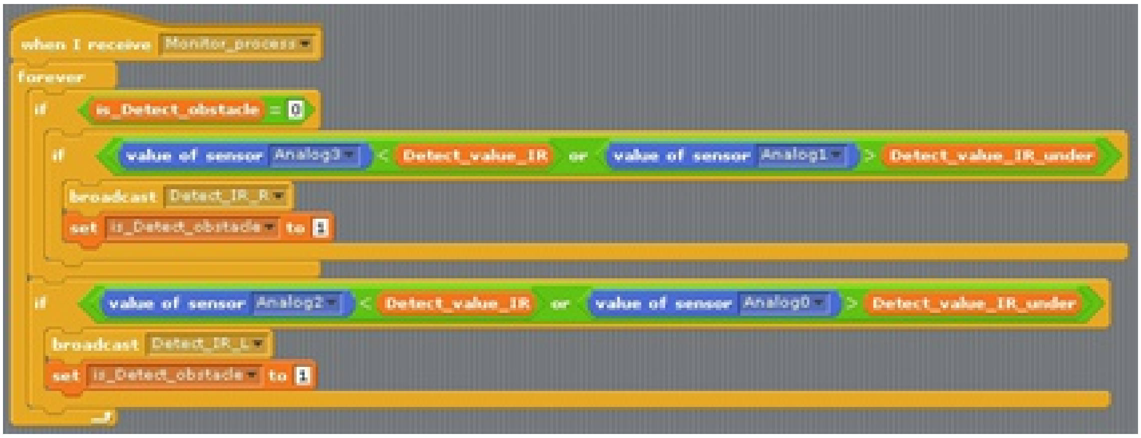
If the infrared sensor connected to the left side of the car detects an obstacle, the car must move back and go to the right, opposite of the obstacle. Likewise, if an infrared sensor connected to the right side of the car detects an obstacle, the car must move back and then to the left, which is the opposite direction of the obstacle. And, in each situation, the variable that judges the obstacle must change to '1'. Create a script for infrared sensor recognition.

Commentary

Use the IR sensor to judge after recognizing the situation

If the infrared sensor connected to the right front is smaller than the reference value or if the infrared sensor connected at the lower right is larger than the reference value, the car should operate in the backward direction and the steering wheel in the left direction. Likewise, if the infrared sensor connected to the left front is smaller than the reference value, or if the infrared sensor connected at the bottom left is larger than the reference value, the car should operate in the backward direction and the steering wheel in the right direction.

First, there must be a condition that recognizes the situation when the situation recognition variable is '0'. Then you have to decide which obstacle is detected by which infrared sensor. Please create the script below



```
when I receive Monitor_process
forever
    if is_Detect_obstacle = 0
        if value of sensor Analog3 < Detect_value_IR or value of sensor Analog1 > Detect_value_IR_under
            broadcast Detect_IR_R
            set is_Detect_obstacle to 1
    if value of sensor Analog2 < Detect_value_IR or value of sensor Analog0 > Detect_value_IR_under
        broadcast Detect_IR_L
        set is_Detect_obstacle to 1
```

The variable 'is_Detect_Obstacle' is a variable used to recognize obstacles. If the value of this variable is '0', it recognizes the obstacle. If it is '1', it judges the situation and acts. At the end of the operation, we need to change the value from '1' to '0' again. By repeating this process, you become aware of obstacles.

Therefore, when the right infrared sensor recognizes an obstacle, it is necessary to change the value of the variable to '1' after performing the judgment operation. Likewise, if the left infrared sensor recognizes an obstacle, you should change the value of the variable to '1' after performing the judgment action.

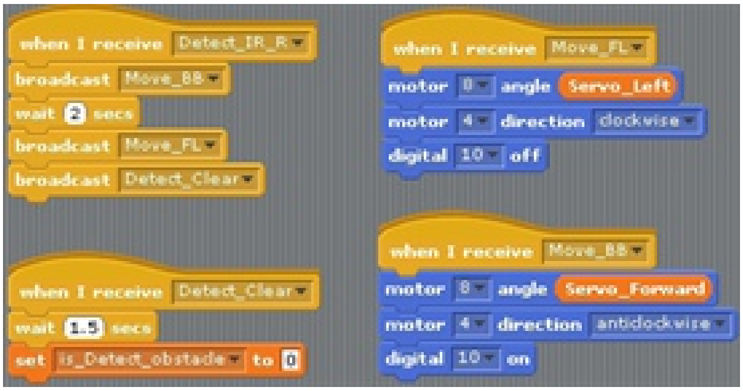
Here, the judgment operation is to operate in the opposite direction. When the left infrared sensor is recognized, it should go to the right, and when the right infrared sensor is recognized, it should go to the left.

Activities

Operation after judging using right infrared sensor

In the 'Monitor_process' script created on the previous page, if the infrared sensor connected to the right recognizes an obstacle, it broadcasts 'Detect_IR_R' and stores '1' in the variable 'is_Detect_obstacle'.

Now, let's code what the script 'Detect_IR_R' should do. This script shows the situation when the right infrared sensor is recognized, so the car must go to the left. At this time, instead of going straight to the left, you have to go back to the left after the sound is heard. Create the scripts below!



```
when I receive Detect_IR_R
broadcast Move_BB
wait 2 secs
broadcast Move_FL
broadcast Detect_Clear

when I receive Detect_Clear
wait 1.5 secs
set is_Detect_obstacle to 0

when I receive Move_FL
motor 0 angle Servo_Left
motor 4 direction clockwise
digital 10 off

when I receive Move_BB
motor 8 angle Servo_Forward
motor 4 direction anticlockwise
digital 10 on
```

If you receive a broadcast called 'Detect_IR_R', broadcast 'Move_BB' and wait for 2 seconds. Then, broadcasts 'Move_FL' and 'Detect_Clear'. I do not know if you've noticed, but here 'Move_BB' is the script that makes the car sound and run backwards, and 'Move_FL' is the script that drives the car to the left. When the operation is completed, the 'is_Detect_obstacle' variable is set to '0' through a script called 'Detect_Clear' to recognize the obstacle again.

The 'Move_FL' script will change the servo motor angle to the left, then move the motor to the left by turning on the DC motor, then the 'Move_BB' script will change the servo motor angle to the forward direction and will let the car go back for two seconds.

The 'Detect_Clear' script waits for the car to run for an unobstructed period of 1.5 seconds, then saves a '0' in the 'is_Detect_obstacle' variable to resume obstacle recognition.

CLASS 4

Activities

Use the left infrared sensor to operate after judgment

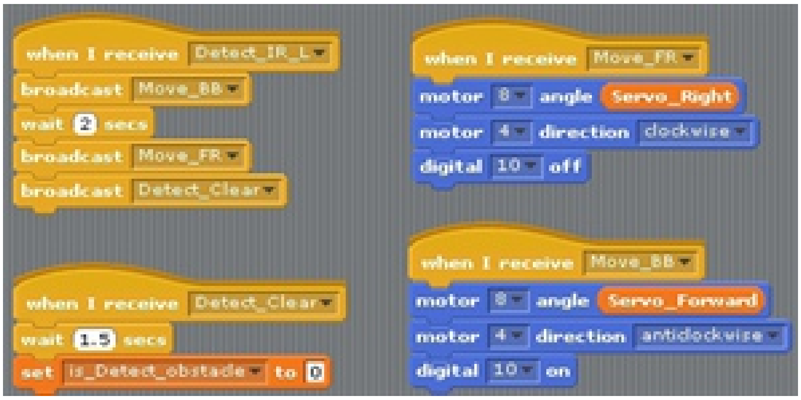
Now, let's code a script for 'Detect_IR_L' broadcast from the 'Monitor_process' script. Please refer to the 'Detect_IR_R' script created on the previous page. This time, it indicates the situation when the left infrared sensor is recognized, so the car must go to the right. At this time, it does not go right instantly, but it has to go back to the right after a little bit. Create a 'Detect_IR_L' script, and if you need other script, create it!

Commentary

Operation after judging using the left IR sensor

In your 'Monitor_process' script, if the infrared sensor connected to the left detects an obstacle, it broadcasts 'Detect_IR_L' and it stores '1' in variable 'is_Detect_obstacle'.

The 'Detect_IR_L' script shows the situation when the left infrared sensor is recognized, so the car must go to the right. At this time, it does not go right to the right but has to go back to the right a little back. Create the scripts below!



```
when I receive Detect_IR_L
broadcast Move_BB
wait 2 secs
broadcast Move_FR
broadcast Detect_Clear

when I receive Detect_Clear
wait 1.5 secs
set is_Detect_obstacle to 0

when I receive Move_FR
motor 8 angle Servo_Right
motor 4 direction clockwise
digital 10 off

when I receive Move_BB
motor 8 angle Servo_Forward
motor 4 direction anticlockwise
digital 10 on
```

If it receives a 'Detect_IR_L' broadcast, broadcasts 'Move_BB' and waits 2 seconds. After that, broadcast 'Move_FR' and 'Detect_Clear'. As before, 'Move_BB' is the script that runs the car back, and 'Move_FR' is the script that runs the car to the right. When the operation is completed, set the 'is_Detect_obstacle' variable to '0' through a script called 'Detect_Clear' to recognize the obstacle again.

The 'Move_FR' script will change the servomotor angle to the right, then move the motor to the right by turning on the DC motor, the 'Move_BB' script will change the servo motor angle to the forward direction and then move the DC motor counterclockwise and let the car go back for two seconds.

The 'Detect_Clear' script waits for the car to run for an unobstructed period of 1.5 seconds, then saves a '0' in the 'is_Detect_obstacle' variable to resume obstacle recognition.

CLASS 4

Activities

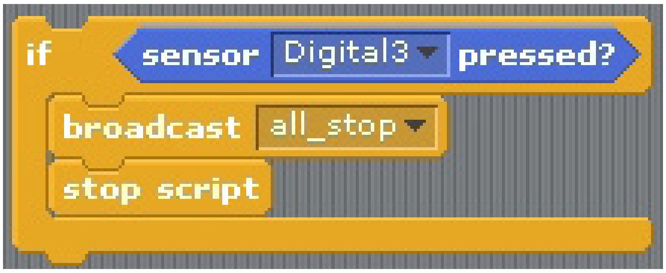
Stop operation using button sensor

When you run the script and press the button sensor, the Tori Autonomous Vehicle will be able to recognize and determine the situation by itself. There are many unexpected situations when you travel like this. At this time, add the code to the existing script to stop te operation by pressing the button sensor. Code all motions to stop when the button sensor is pressed while Tori autonomous vehicle is moving.

Commentary

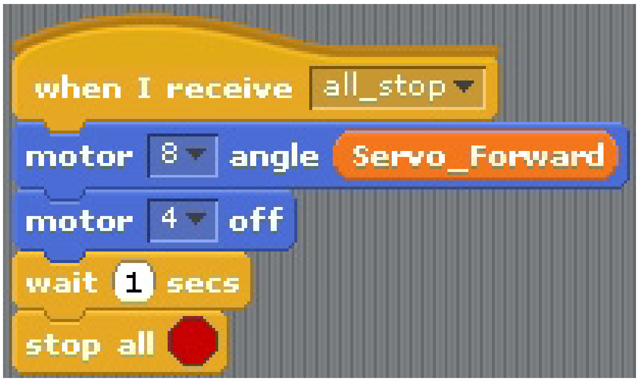
Stop operation using button sensor

If you press the button sensor while the car is running, all operations must be stopped. If so, you can add a conditional statement to the 'Monitor_process' script that provides context awareness. The conditional statement that stops all operation when button sensor is pressed is as below.



```
if <sensor Digital3 pressed?>
    broadcast all_stop
    stop script
```

I think you could easily make the above conditional statements easy enough. This conditional should be included in the 'Monitor_process' script! Press the button sensor, which is a sensor connected to the digital port 3, to broadcast the script 'all_stop' and stop the 'monitor_process' script. The 'all_stop' script puts the steering wheel in the forward direction, stops the DC motor and stops all scripts.



```
when I receive all_stop
motor 8 angle Servo_Forward
motor 4 off
wait 1 secs
stop all
```

The script 'all_stop' is created as above! Place the servo motor angle in the forward direction of the steering wheel and stop the DC motor. Wait for a second and then stop the entire action with the 'Stop All' block, which stops all scripts on the scratch screen.

If you add the two scripts above, you can stop the entire operation with the button sensor!

Activities

Use ultrasonic sensor to judge after recognizing the situation

Now, try to use the ultrasonic sensor to recognize the situation. The closer the distance of the ultrasonic sensor, the smaller the analog value. It can detect obstacles by using the ultrasonic sensor connected to the servo motor.

I will measure the value three times after letting the steering wheel turn to the left and the ultrasonic sensor connected to the servomotor can recognize the situation at the left 45degree angle. With these three measured values, it can averagely measure obstacles more precisely how far from the ultrasonic sensor the obstacle is. After that, you do the same thing by turning the steering wheel to the right.

Here, the measurement of the sensor is called sampling. It is called a filter to find the most accurate value by averaging the sampled values. Above, the ultrasonic sensor samples the value and determines the filter value with the sampled value.

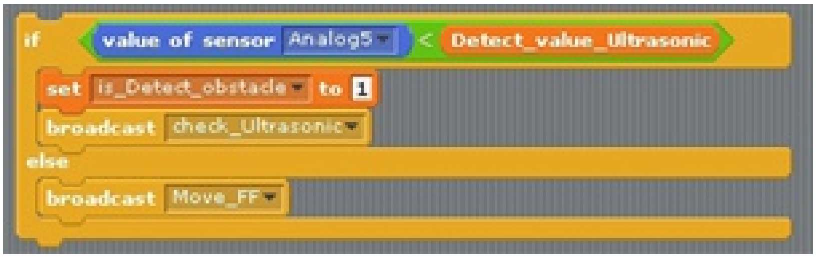
Now let's add a script that recognizes the situation using the ultrasonic sensor in the 'Monitor_process' script. If the value of the ultrasonic sensor is smaller than the predetermined value of the ultrasonic sensor, '1' is stored in the 'is_Detect_obstacle' variable. After the ultrasonic sensor has sampled 3 times after setting the servo motor to the left, the servo motor is set to the right, we need to let the sensor determine the filter value after sampling 3 times.

It can then compare the filter value on the left with the filter value on the right to determine which obstacle is on either side. If neither the infrared sensor nor the ultrasonic sensor is recognized, it should keep going straight, right? Make the scripts described so far in your way!

Commentary

Use ultrasonic sensor to judge after recognizing the situation

If the ultrasonic sensor is smaller than the 'Detect_value_Ultrasonic' parameter, the ultrasonic sensor should sample. However, if the ultrasonic sensor is not recognized and the infrared sensor is not recognized, there is no obstacle, so the robot must keep moving forward. Let's create a script that satisfies these conditions first. Create the script below!

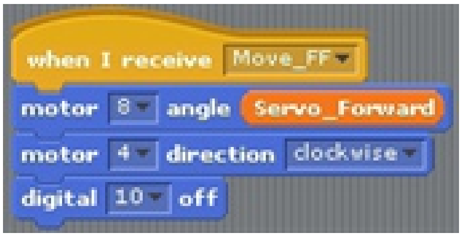


```
if <(value of sensor [Analog5 v]) < (Detect_value_Ultrasonic)>
    set [is_Detect_obstacle v] to [1]
    broadcast [check_Ultrasonic v]
else
    broadcast [Move_FF v]
```

You can put the above script in the conditional statement of the 'Monitor_process' script.

If the ultrasonic transmitter is less than 'Detect_value_Ultrasonic', it will broadcast 'check_Ultrasonic' which is a script to save '1' to 'is_Detect_obstacle' variable and to sample using ultrasonic sensor.

Otherwise, that is, if no sensor is recognized, there is no obstacle, so the car must continue straight ahead. Therefore, you have to turn the steering wheel in the forward direction and then operate the DC motor. So, let 's make' Move_FF 'and let the car move forward. The 'Move_FF' script looks like this:



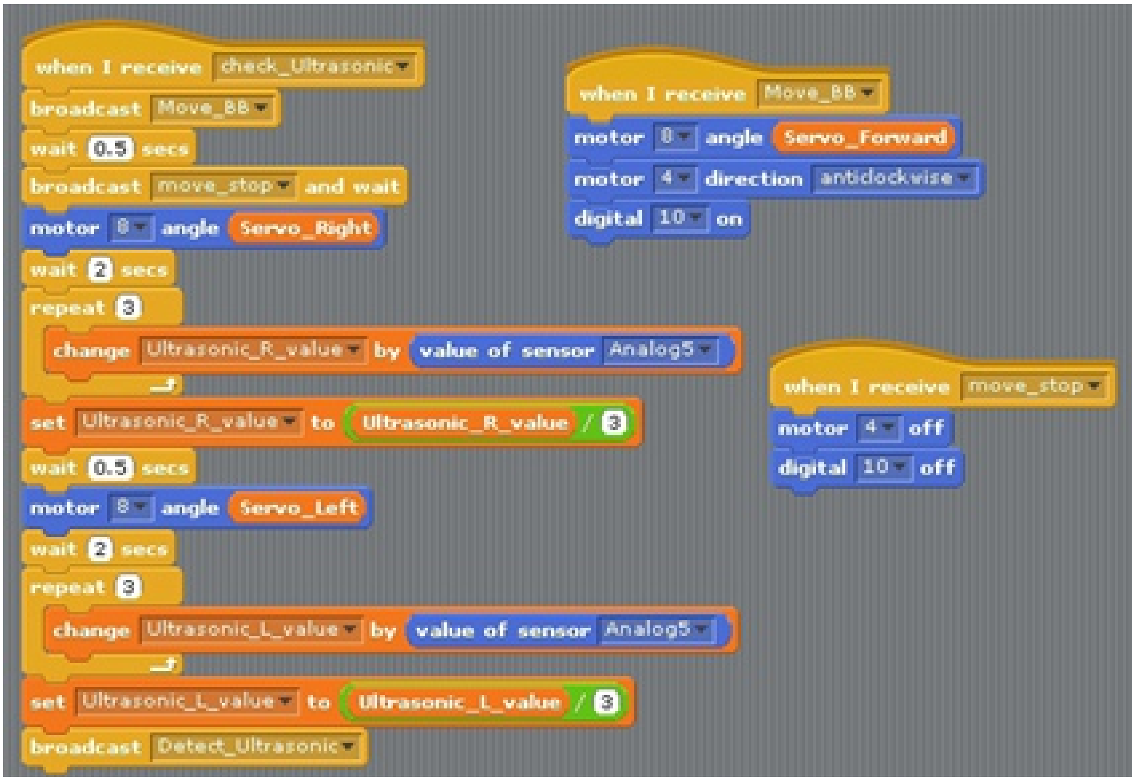
```
when I receive [Move_FF v]
motor [8 v] angle (Servo_Forward)
motor [4 v] direction [clockwise v]
digital [10 v] off
```

Now, let's create a 'check_Ultrasonic' script that allows the ultrasonic sensor to sample! The ultrasonic sensor samples the left direction three times, averages it down, sets the filter value, and the average of the three sampling times to the right, decides the filter value, and codes it to judge where the object is!

Commentary

Sampling with an ultrasonic sensor

Now let's create a script called 'check_Ultrasonic' that we broadcasted earlier. Compare it to the script you've created and see what's different! The scripts needed for sampling the ultrasonic sensors are shown below. Go ahead and make it!



```
when I receive check_Ultrasonic
broadcast Move_BB
wait 0.5 secs
broadcast move_stop and wait
motor 8 angle Servo_Right
wait 2 secs
repeat 3
    change Ultrasonic_R_value by value of sensor Analog5
set Ultrasonic_R_value to Ultrasonic_R_value / 3
wait 0.5 secs
motor 8 angle Servo_Left
wait 2 secs
repeat 3
    change Ultrasonic_L_value by value of sensor Analog5
set Ultrasonic_L_value to Ultrasonic_L_value / 3
broadcast Detect_Ultrasonic
```

```
when I receive Move_BB
motor 8 angle Servo_Forward
motor 4 direction anticlockwise
digital 10 on
```

```
when I receive move_stop
motor 4 off
digital 10 off
```

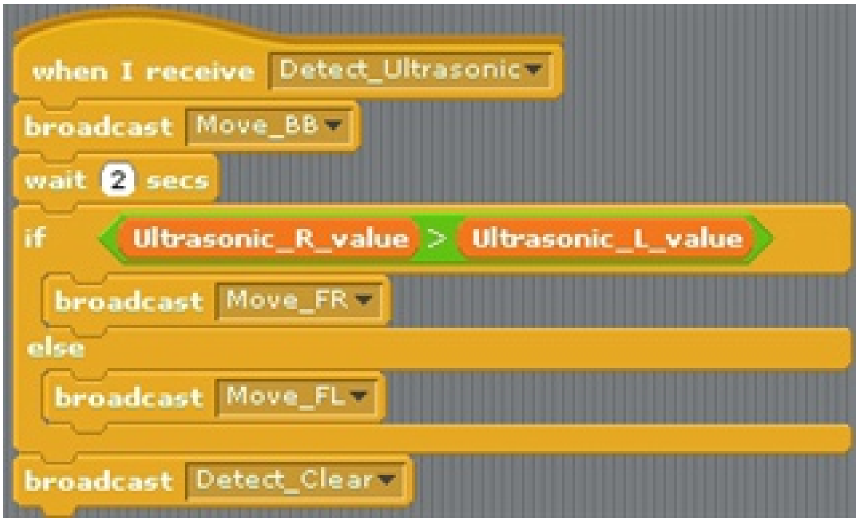
When it gets 'check_Ultrasonic', it runs backwards for half a second, and broadcast 'move_stop' and wait. After that, it starts the sampling with the servo motor angle right. The ultrasonic sensor value at this time is accumulated three times in the 'Ultrasonic_R_value' variable. Then divide the accumulated value in this variable by 3, average it down, and save it back in the 'Ultrasonic_R_value' variable.

So, this variable stores the value that the ultrasonic sensor samples the rightward direction three times and the average is reduced. The 'Ultrasnoic_R_value' value at this time will be the filter value in the right direction. Now, with the servo motor angle left to the same exactitude, broadcast the "Detect_Ultrasonic" script to determine how the action should be done.

Activities

Operation after judgment using ultrasonic sensor

In the 'check_Ultrasonic' script that we created on the previous page, let's create a 'Detect_Ultrasonic' script that was broadcast to determine the ultrasound sensor value at the end. This script is very simple. This is because the filter value of the ultrasonic sensor is compared with the filter value of the ultrasonic sensor in the left direction and the filter value of the ultrasonic sensor is measured in the right direction. Please create the script below.



```
when I receive Detect_Ultrasonic
broadcast Move_BB
wait 2 secs
if Ultrasonic_R_value > Ultrasonic_L_value
    broadcast Move_FR
else
    broadcast Move_FL
broadcast Detect_Clear
```

If it receives a broadcast called 'Detect_Ultrasonic', broadcast 'Move_BB' and wait for 2 seconds. Then compare the values of 'Ultrasonic_R_value' and 'Ultrasonic_L_value'. If the value of 'Ultrasonic_R_value' is larger, the filter value of the ultrasonic sensor on the right side is larger than that of the ultrasonic sensor on the left side, so it can be judged that the obstacle is closer to the left side. So, the car should go to the right, so we have to broadcast the 'Move_FR' script.

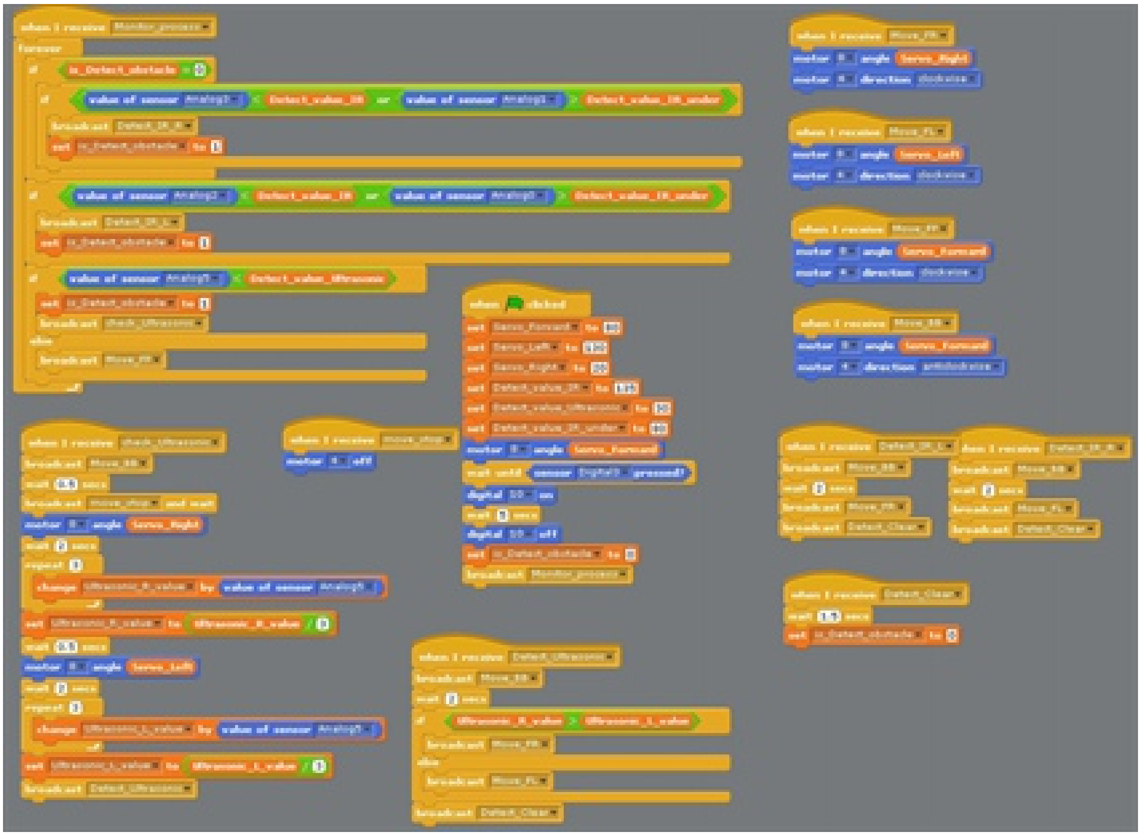
Conversely, if the value of 'Ultrasonic_L_value' is larger, the filter value of the ultrasonic sensor on the left side is larger than the filter value of the ultrasonic sensor on the right side, so it can be judged that the obstacle is closer to the right side. So, it has to go to the left side of the car, so it can broadcast the 'Move_FL' script.

After the judgment is finished, it should broadcast 'Detect_Clear' script and make 'is_Detect_obstacle' variable '0' so that it can recognize the obstacle again.

Activities

Tori Autonomous Vehicle coding

Combine all the scripts you've created so far, press the Run Script button, press the button sensor and it will work!



```
when I receive Monitor_process
forever
  if is_Detect_obstacle = 0
    if value of sensor Analog1 < Detect_value_IR or value of sensor Analog1 > Detect_value_IR_under
      broadcast Detect_IR_R
      set is_Detect_obstacle to 1
  if value of sensor Analog2 < Detect_value_IR or value of sensor Analog2 > Detect_value_IR_under
    broadcast Detect_IR_L
    set is_Detect_obstacle to 1
  if value of sensor [illegible] < Detect_value_Ultrasonic
    set is_Detect_obstacle to 1
    broadcast check_Ultrasonic
  else
    broadcast Move_FF
```

```
when clicked
set Servo_Forward to [illegible]
set Servo_Left to [illegible]
set Servo_Right to [illegible]
set Detect_value_IR to [illegible]
set Detect_value_Ultrasonic to [illegible]
set Detect_value_IR_under to [illegible]
motor [illegible] angle Servo_Forward
wait until sensor [illegible] pressed?
digital [illegible] on
wait 1 secs
digital [illegible] off
set is_Detect_obstacle to 0
broadcast Monitor_process
```

```
when I receive check_Ultrasonic
broadcast Move_BB
wait 0.5 secs
broadcast move_stop and wait
motor [illegible] angle Servo_Right
wait 2 secs
repeat [illegible]
  change Ultrasonic_R_value by value of sensor [illegible]
set Ultrasonic_R_value to Ultrasonic_R_value / [illegible]
wait 0.5 secs
motor [illegible] angle Servo_Left
wait 2 secs
repeat [illegible]
  change Ultrasonic_L_value by value of sensor [illegible]
set Ultrasonic_L_value to Ultrasonic_L_value / [illegible]
broadcast Detect_Ultrasonic
```

```
when I receive move_stop
motor [illegible] off
```

```
when I receive Detect_Ultrasonic
broadcast Move_BB
wait 1 secs
if Ultrasonic_R_value > Ultrasonic_L_value
  broadcast Move_FR
else
  broadcast Move_FL
broadcast Detect_Clear
```

```
when I receive Move_FR
motor [illegible] angle Servo_Right
motor [illegible] direction clockwise
```

```
when I receive Move_FL
motor [illegible] angle Servo_Left
motor [illegible] direction clockwise
```

```
when I receive Move_FF
motor [illegible] angle Servo_Forward
motor [illegible] direction clockwise
```

```
when I receive Move_BB
motor [illegible] angle Servo_Forward
motor [illegible] direction anticlockwise
```

```
when I receive Detect_IR_L
broadcast Move_BB
wait 2 secs
broadcast Move_FR
broadcast Detect_Clear
```

```
when I receive Detect_IR_R
broadcast Move_BB
wait 2 secs
broadcast Move_FL
broadcast Detect_Clear
```

```
when I receive Detect_Clear
wait 1.5 secs
set is_Detect_obstacle to 0
```

The whole script can be found at http://www.robotori.com/web_eng -> Moretips -> Manual ->EDU ->Logic boost CODING CLASS 4 ->download whole script

SCRATCH CODING KIT

Logic Boost

Kart Rider

LESSON 8

CLASS 4

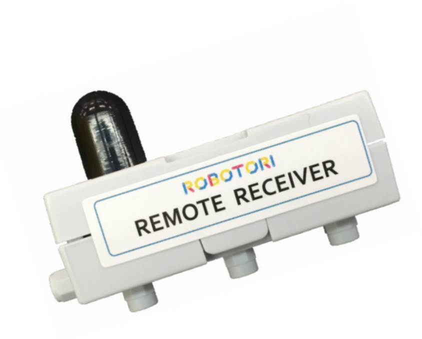
Making Robot

Kart Rider

Before you do any further coding, let's quickly learn how to manipulate the model using a wireless remote control. Knowing this way, you can create the model you want and control it with a wireless remote control.

The wireless remote control can control the robot by using the light called infrared light. Infrared refers to particles that are invisible to sunlight. So, using infrared rays to control the robot is called infrared communication. It is common in TV remote controls and mobile phones we watch. When I send the infrared rays from the remote control, the infrared ray sensor receives the infrared signal and the robot moves. Therefore, when you control the robot with a wireless remote control, you must steer towards the remote sensor to the sensor that detects the signal.

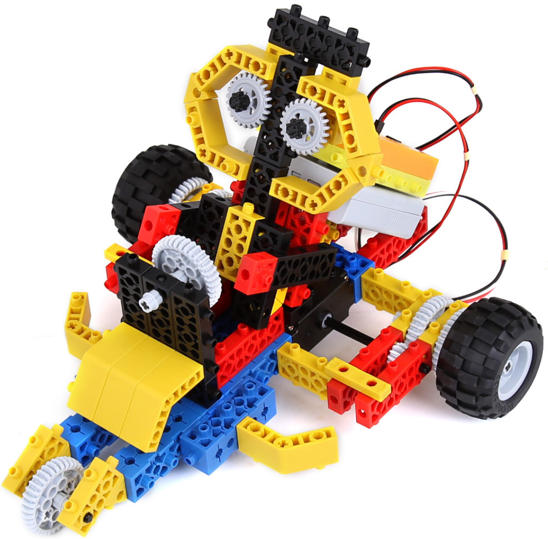
Now, let's move on to make a Kart Rider.



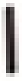
Making Robot

Kart Rider

It is an image of the shape of Kart Rider, a racing car game. Create a variety of cart riders to race your robots.



Kart Rider Material list

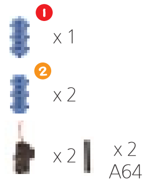
Diamond H8 x 3	Diamond H6 x 5	Rubi 8 x 2	Rubi 7 x 4	Rubi 4 x 4	Rubi 6 x 6	Rubi 2 x 3	Rubi 0 x 4	Mini 2 x 13	Mini 1 x 3	Curve x 10	Link x 4	Triangle x 8
												
Sawtooth 8 x 2	Sawtooth 12 x 2	Crown x 2	Sawtooth 36 x 4	Middle connector x 11	Off -road wheel x 2	A 45 x 4	A 64 x 4	Battery 6V Case x 1	DCMotor x 2	REMOTE RECEIVER x 1	Mainboard x 1	connector x 2

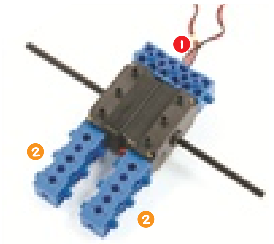
CLASS 4

Making a Robot

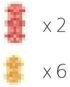
Making Kart Rider

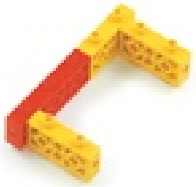
1 **1-1**





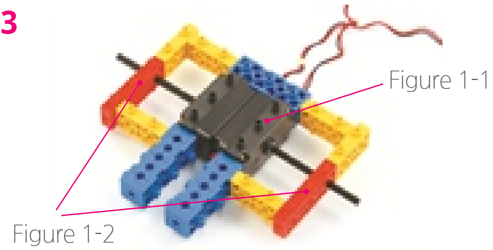
1-2



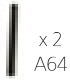


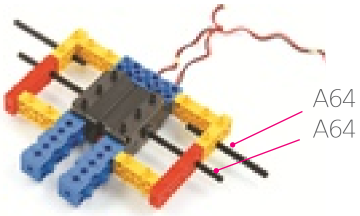
Make two identical models

1-3



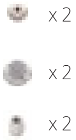
1-4

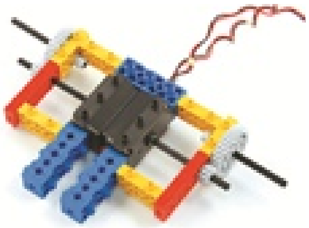




Assemble it like the image on the other side

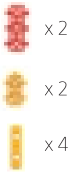
2 **2-1**

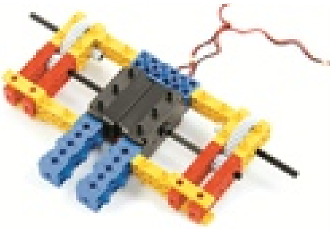




Assemble it like the image on the other side

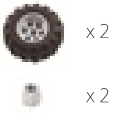
2-2

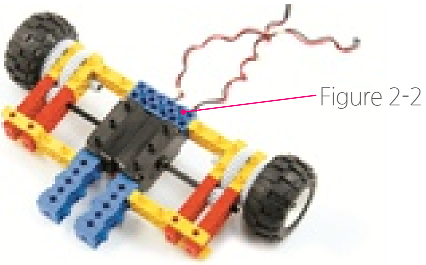




Assemble it like the image on the other side

2-3





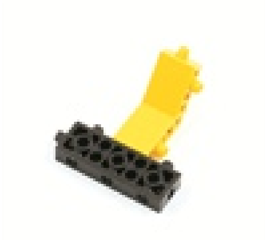
Making a Robot

Making Kart Rider

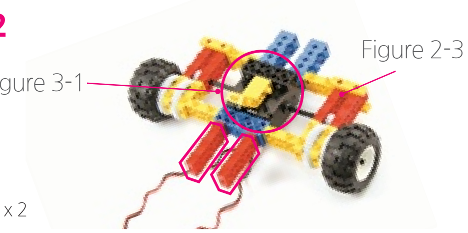
3 **3-1**

x 1

x 1



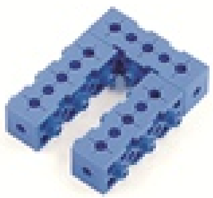
3-2



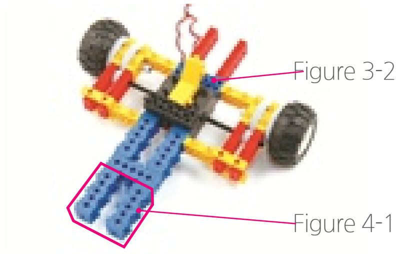
x 2

4 **4-1**

x 3



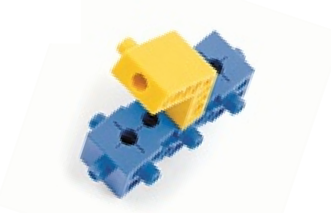
4-2



5 **5-1**

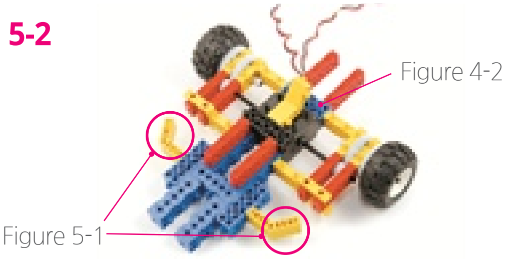
x 2

x 2



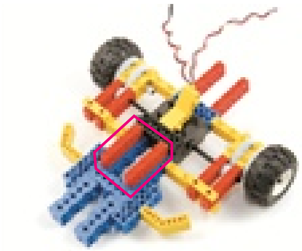
Make two identical models

5-2



5-3

x 2

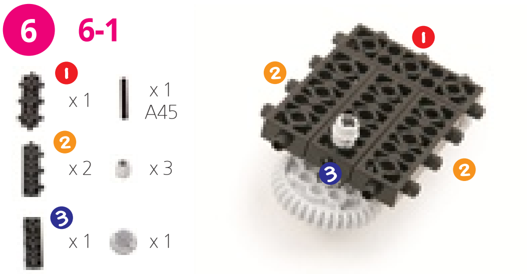


Assemble with 5-2 model

Making Kart Rider

6

6-1



6-2

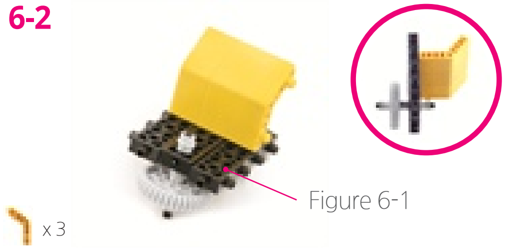


Figure 6-1

6-3

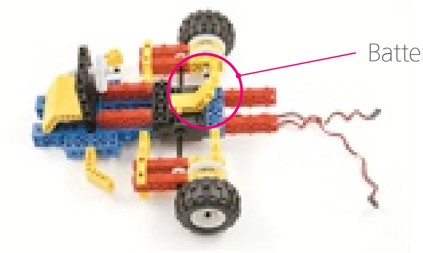
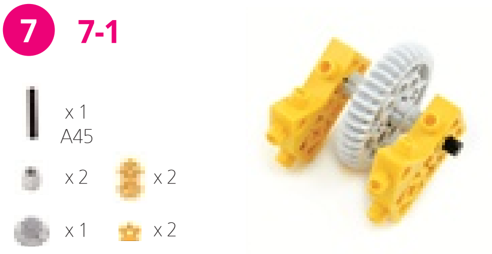


Figure 14-5
Battery case assembly location

Assemble the models you made in picture 5-3 and 6-2

7

7-1



7-2

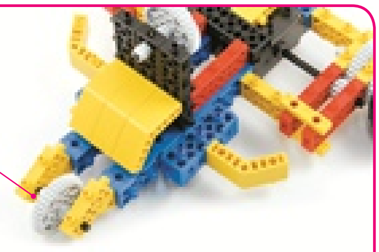
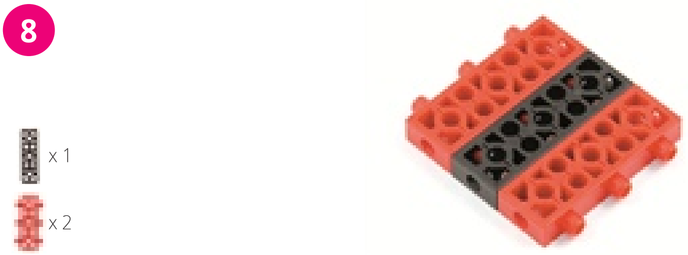


Figure 7-1

Assemble the models you made in picture 6-3 and 7-1

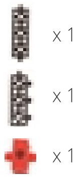
8

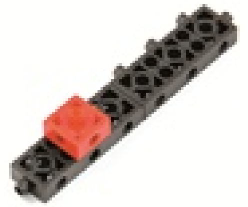


Making a Robot

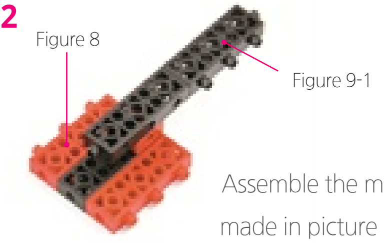
Making Kart Rider

9 9-1



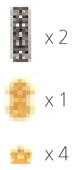


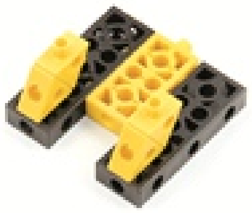
9-2



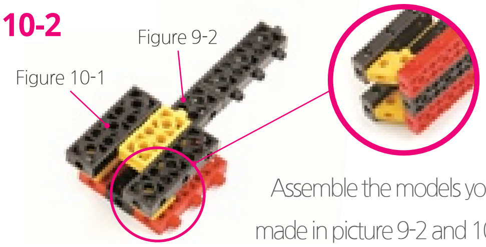
Assemble the models you made in picture 8 and 9-1

10 10-1



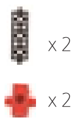


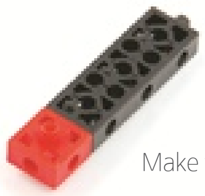
10-2



Assemble the models you made in picture 9-2 and 10-1

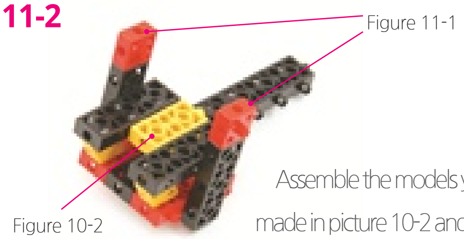
11 11-1





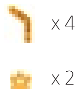
Make two identical models

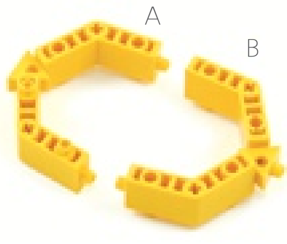
11-2



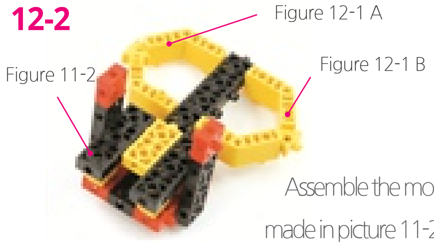
Assemble the models you made in picture 10-2 and 11-1

12 12-1





12-2



Assemble the models you made in picture 11-2 and 12-1

CLASS 4

Making a Robot

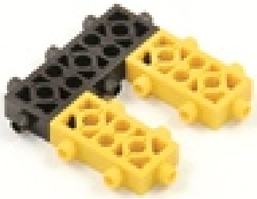
Making Kart Rider

13

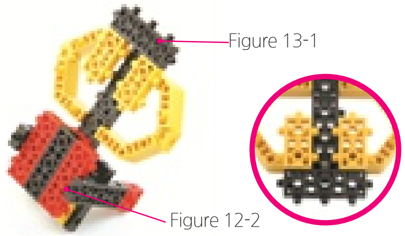
13-1

x 1

x 2



13-2



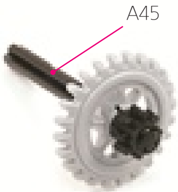
14

14-1

x 2
A45

x 2

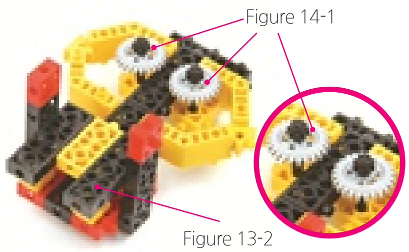
x 2



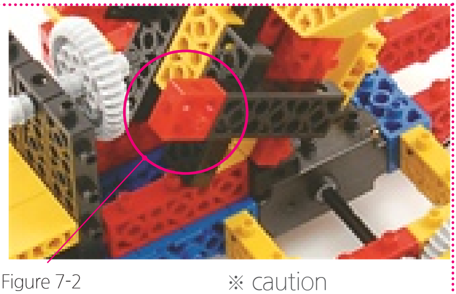
Make two identical models

14-2

x 2

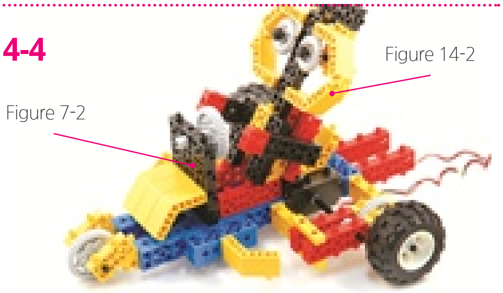


14-3



※ caution

14-4

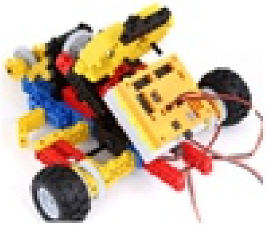


14-5

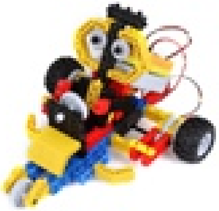
x 1

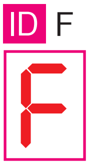
x 1

x 1



14-6



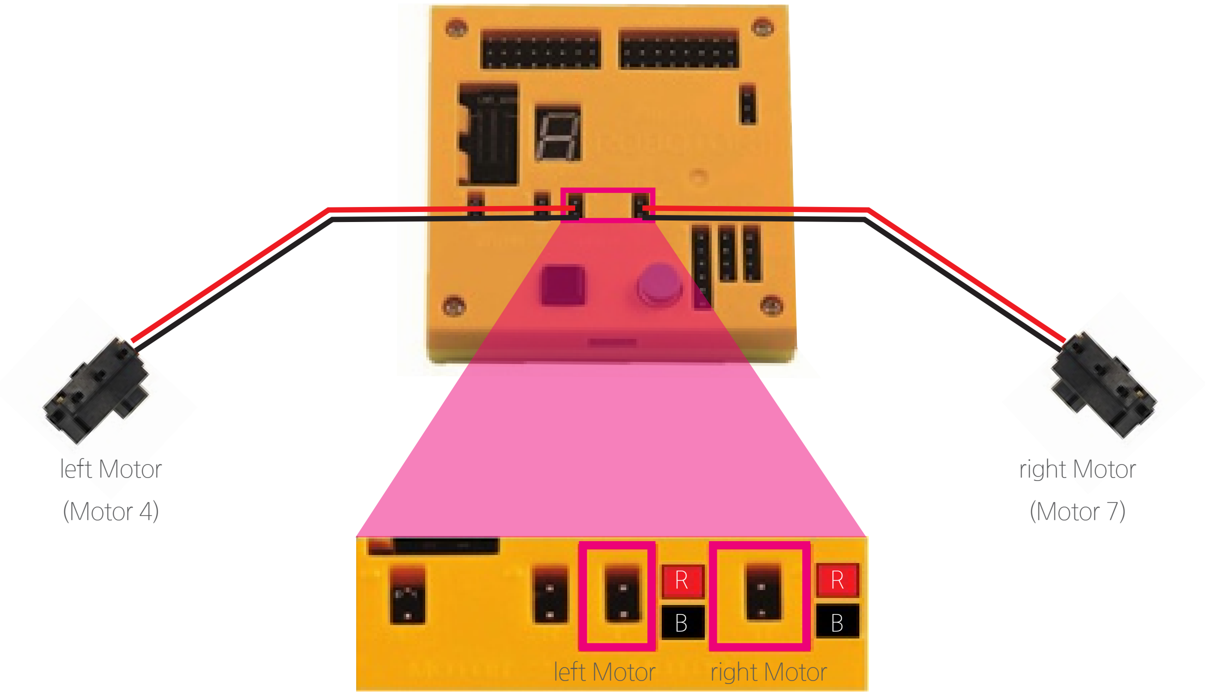


※ Setting the operation mode

Connecting main cell

Connecting Kart Rider

How should we connect the Kart Rider model with the main cell?
Look at the picture below closely. When connecting the motor to the left side of the cart rider, be sure to insert the red line on the upper pin and the black line on the lower pin firmly. When connecting the right motor, please connect the red line to the upper pin and the black line to the lower pin firmly.



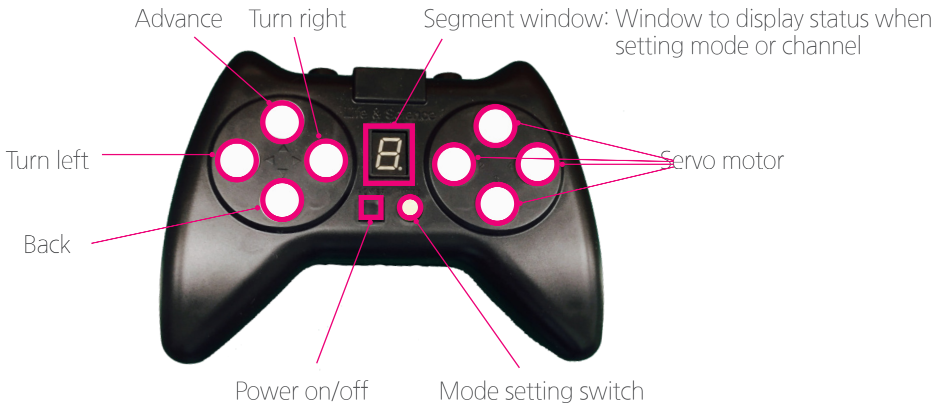
※Carefully connect the wires.

CLASS 4

Wired/Wireless remote control

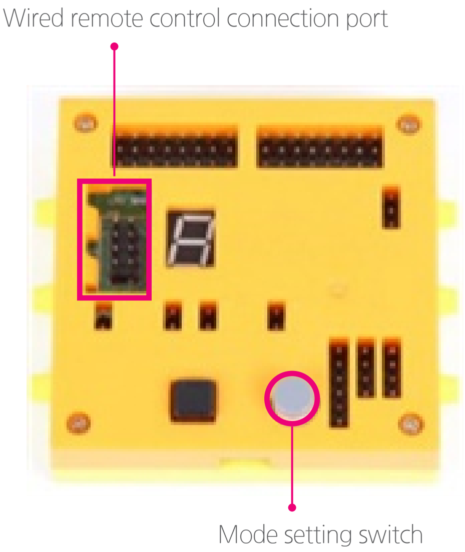
Using Wired/wireless remote control

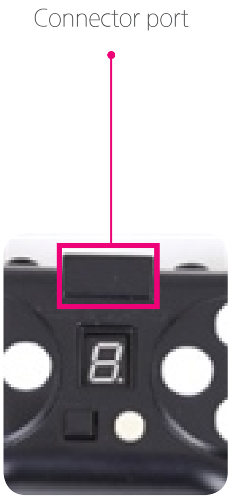
The wired / wireless remote control can be used as a wired remote control by plugging in a connector or by using a wireless remote control together with a remote receiver after disconnecting the connector

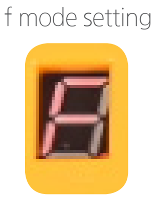


1. Using as wired remote control

Press 'All' change switch of main board continuously to set 'F' mode and connect remote controller to main board





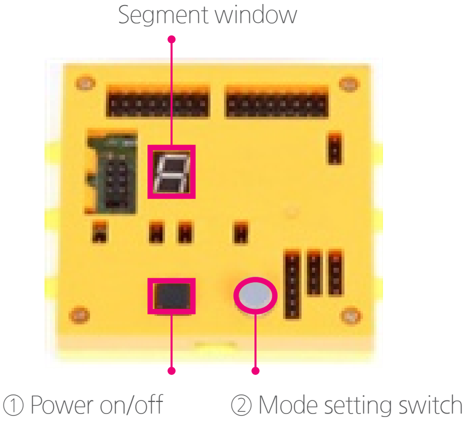


Wired/Wireless remote control

2. Using a wireless remote control

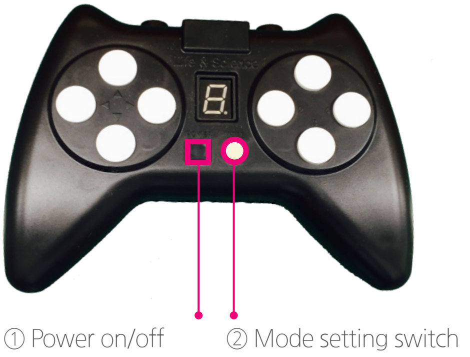
When using a wireless remote, there are few things you have to set up before using.

1) Setting the same channel as the main board and remote control
2) Connect REMOTE RECEIVER and the main board (wireless remote control and REMOTE RECEIVER can exchange signal and control motion)
3) Set main board to 'F' mode.



How to set the main board channel

1. Switch 1 is turned off
2. Turn on switch 1 while pressing switch 2
3. Segment window shows the currently saved channel
4. Change channel by pressing switch 2 again.
5. Move to desired channel and wait 2 seconds. IF segment blinks 3 times, channel setting is completed.



How to set wireless remote control channel

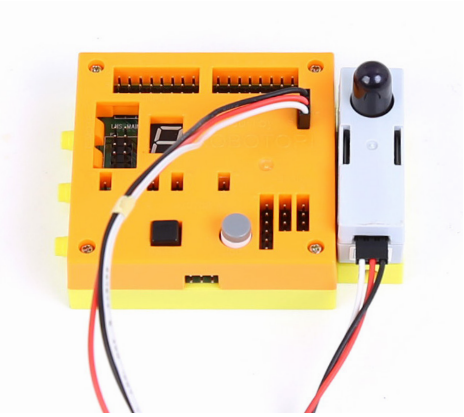
1. Switch 1 is turned off
2. Turn on switch 1 while pressing switch 2
3. Segment window shows the currently saved channel
4. Press switch 1 and 3 (top and bottom) to change channel
5. Move to desired channel and press switch 4 to save.

CLASS 4

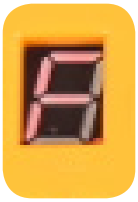
Wired/Wireless remote control

Using wireless remote control

Did you set both the main board and the wireless remote control to same channel?
Now let's connect REMOTE RECEIVER and main board.



Connect the connection jacks in the order of black, red, and white as shown on the left. Be careful not to connect them in opposite order!



Set as F mode

Finally, press the mode change switch continuously to set it to 'F' mode.

Controlling Kart Rider

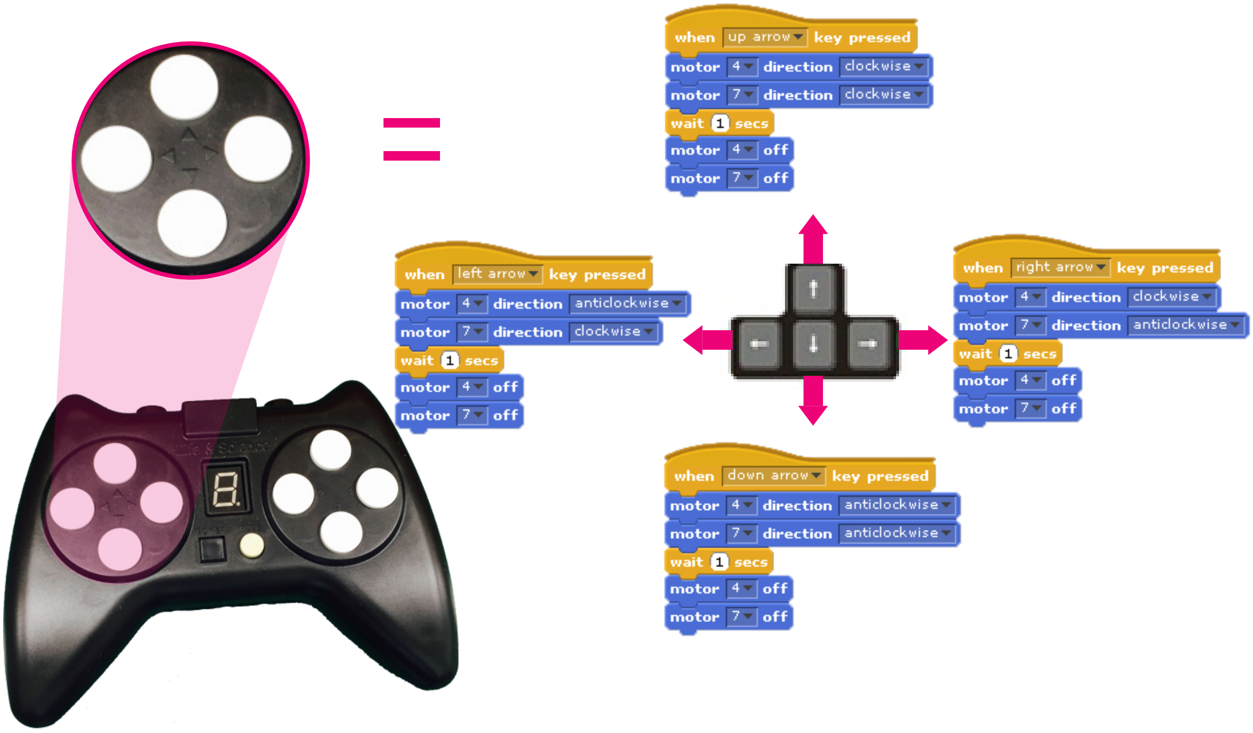
Controlling Kart Rider

Now that you've set up your wireless remote control, you can fly your cart rider! The wireless remote control gives the advantage of controlling the robot without connecting it to the computer.

However, there are limits to what a robot can do when you think about it. The wireless remote control can only control the operation of the robot! If you want to see the movement of the robot easily, you should use the wireless remote control.

The picture below is a comparison of the wireless remote control with the one you coded in the scratch program. In the picture below, you can see how the buttons on the wireless remote control are coded.

Now go ahead and control the kart rider!



```
when up arrow key pressed
motor 4 direction clockwise
motor 7 direction clockwise
wait 1 secs
motor 4 off
motor 7 off
```

```
when left arrow key pressed
motor 4 direction anticlockwise
motor 7 direction clockwise
wait 1 secs
motor 4 off
motor 7 off
```

```
when right arrow key pressed
motor 4 direction clockwise
motor 7 direction anticlockwise
wait 1 secs
motor 4 off
motor 7 off
```

```
when down arrow key pressed
motor 4 direction anticlockwise
motor 7 direction anticlockwise
wait 1 secs
motor 4 off
motor 7 off
```

MEMO

FCC Information to User

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Caution

Modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

FCC Compliance Information : This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

This equipment complies with FCC radiation exposure limits set forth for an uncontrolled environment.