Studuino

Programming Environment

Manual

Published 2014/11/01
Revised 2018/10/01

Version History

Date	Content
2014/11/01	First version
2017/01/16	Updated for new Studuino website
2017/03/28	Revised with added features for Windows tablets
2017/04/14	Changed access guide based on new Studuino website
2017/10/31	Updated version 1.5 of the Studuino software
2018/02/01	Revised to include screenshots for the Icon Programming Environment’s new tablet-friendly GUI and new Block Programming Environment features
2018/03/01	Revisions to the Block Programming Environment's Motor Calibration GUI
2018/05/07	Added extra blocks
2018/08/22	Added extra blocks
2018/10/01	Added Data Logger

Index

1. Getting Started

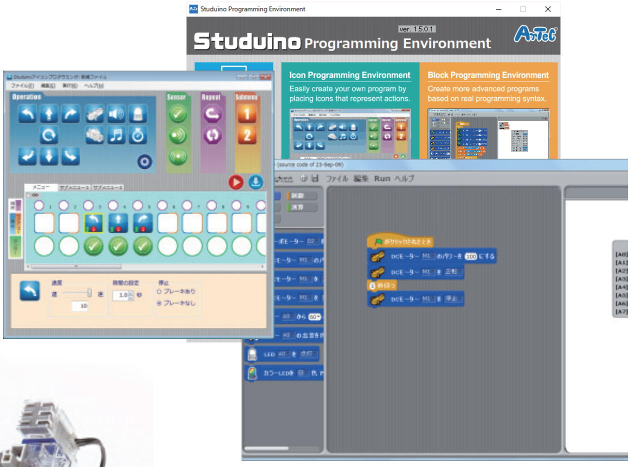
You can download every manual listed here from the Studuino Software website. Follow the steps below to download them:

1) Go to the **Studuino Software Downloads** page at http://artec-kk.co.jp/studuino/en.
2) Mouse over the **Software** section and click **Studuino**.
★ The page you'll need to visit depends on the set or electronic parts you own. See the website for more details.
3) Now download the manual you need under **Instruction Manuals** or **Guides**.

The information in this manual is subject to revision at any time.

2. ArtecRobo and the Studuino Programming Environment

2.1. Overview and Features



Studuino Programming Environment

ver. 1.5.0.1

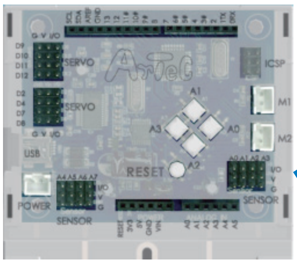
Studuino Programming Environment

Icon Programming Environment

Easily create your own program by placing icons that represent actions.

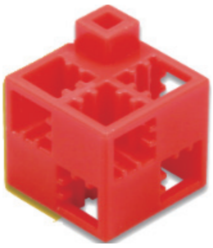
Block Programming Environment

Create more advanced programs based on real programming syntax.

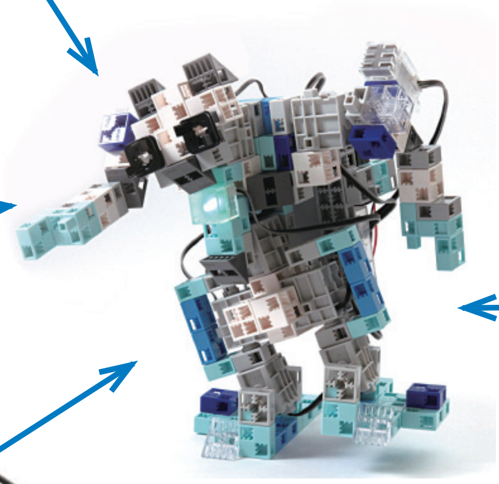


Arduino Compatible

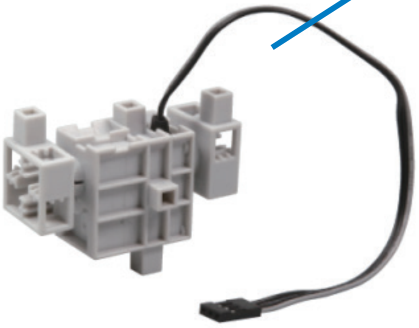
Studuino



ArTeC Blocks



Studuino Programming Environment



Studuino

Motors, LEDs, Buzzers and Sensors

Combine blocks and parts to build your robot

Program your robot using programming environments made for beginners and experts

ArtecRobo is a series that includes Studuino, Artec blocks, electronic parts, and the Studuino Programming Environment. It allows the user to combine these different elements to create their own original robot.

The Studuino Programming Environment allows you to program by dragging and dropping icons or blocks, making it incredibly easy to create an advanced program for a robot even if you're a complete beginner. The programs you'll make can also be translated into the Arduino programming language right inside of the Studuino Programming Environment, allowing you to unlock the full potential of your robot in Arduino IDE.

2.2. System Requirements

● **Windows**

OS	10 / 8.1 / 7 Vista / XP (32 bit / 64 bit)
Processor (CPU)	Pentium 4, 2 GHz or higher (or equivalent) recommended
Memory	256 MB or higher
USB	USB 2.0 port
Software	Requires Microsoft .NET Framework 4.5 (this is automatically installed along with the Studuino software)
Display	XGA (1024×768) or higher

● **Mac** (only supports the Block Programming Environment)

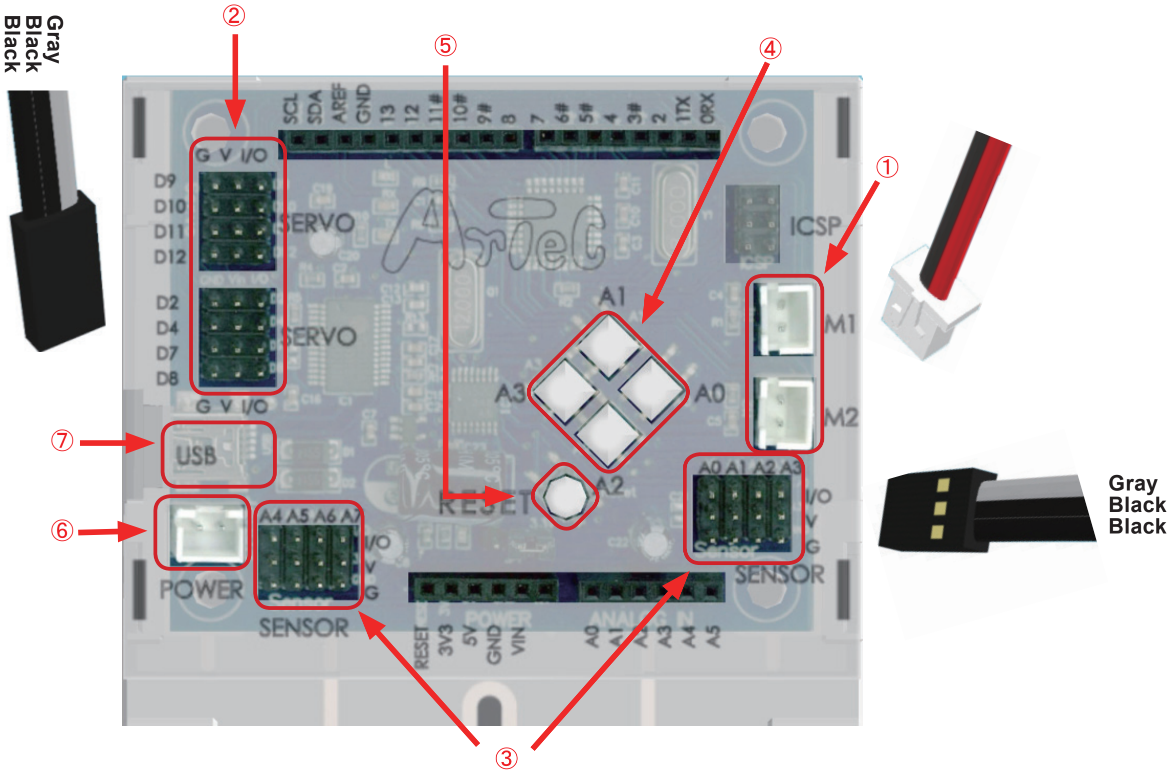
OS	Mac OS X 10.6 to 10.13
Hardware	Minimum OS requirements (Visit Apple's website for details.)
USB	USB 2.0 port

● **Raspberry Pi** (only supports the Block Programming Environment)

OS	Raspbian
Hardware	Raspberry Pi series
USB	USB 2.0 port

2.3. About Studuino

Here you can find out more about the Studuino ports you'll use to make your robots.



① DC Motor Connectors

Use these to connect your DC Motors. Up to two DC Motors can be connected to M1 and M2.

② Servomotor Connectors (Multiple-pin Digital Connector)

You can connect up to eight of them to D2, D4, D7, D8, D9, D10, D11, and D12. The gray signal wire of your Servomotor must face towards the inside of the Studuino.

③ Sensor / LED / Buzzer Connectors (Multiple-pin Analog)

Use these to connect sensors, Buzzers, and LEDs. They're assigned to A0-A7. The gray signal wire of your sensor, LED, or Buzzer must face towards the inside of the Studuino.

Each of your electronic parts uses different types of connectors. Turn to the next page for details.

A ○ shows that the part can be connected to the port

	A0	A1	A2	A3	A4	A5	A6	A7
LED	○	○	○	○	○	○		
Buzzer	○	○	○	○	○	○		
Touch Sensor	○	○	○	○	○	○		
Light Sensor	○	○	○	○	○	○	○	○
Sound Sensor	○	○	○	○	○	○	○	○
IR Photoreflector	○	○	○	○	○	○	○	○
Accelerometer					○	○		

④ Push-buttons

These are the buttons on your Studuino. They're assigned to A0-A3. When you use the Push-buttons, connectors A0-A3 can't be used for sensors, Buzzers, or LEDs.

⑤ RESET Button

Use this button to reset your Studuino. It can come in handy when you need to format or you find that your Studuino isn't working correctly.

⑥ Power Connector (External power supply connector)

Use this to connect your Battery Box, which you'll need to power your DC Motors and Servomotors. You'll also need to use it when your Studuino isn't connected to your PC with a USB cable.

⑦ Communication Jack (USB mini-B)

This connector allows your Studuino to connect to your PC. You can use any commercially available USB cable in addition to the one included.

Keep in mind that the following connectors and buttons can't be used simultaneously:

- DC Motor connector M1 and Servomotor connectors D2, D4
- DC Motor connector M2 and Servomotor connectors D7, D8
- Push-button A0 and Sensor/LED/Buzzer connector A0
- Push-button A1 and Sensor/LED/Buzzer connector A1
- Push-button A2 and Sensor/LED/Buzzer connector A2
- Push-button A3 and Sensor/LED/Buzzer connector A3

Use a DC Motor on M1, for example, and any Servomotors you have connected to D2 or D4 won't work properly. Conversely, using Servomotors on D2 or D4 means that a DC Motor on M1 won't work either.

3. Introduction

You'll need to install both a USB device driver and the Programming Environment software in order to use the Studuino Programming Environment.

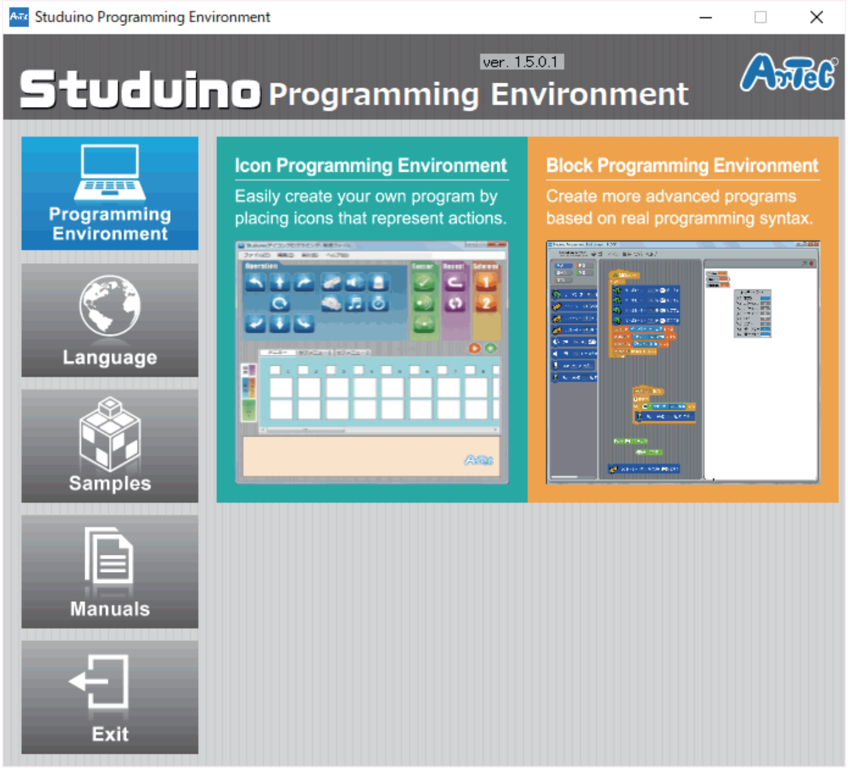
3.1. Installing and Uninstalling the Studuino Programming Environment

You can use the Studuino Programming Environment on Windows, Mac OS X, and Raspberry Pi. Read the **Installing Studuino Software** manual for details on how to install and uninstall the software.

3.2. Installing USB Device Drivers

You'll need to install the device driver before you start programming your Studuino on your PC. Read the **Studuino Setup Guide (Installing USB Device Drivers)** for instructions on how to do this.

4. The Startup Screen



Studuino Programming Environment
ver. 1.5.0.1
Studuino Programming Environment
ArTeC
Programming Environment
Language
Samples
Manuals
Exit
Icon Programming Environment
Easily create your own program by placing icons that represent actions.
Block Programming Environment
Create more advanced programs based on real programming syntax.

Only Windows users can open the Environments using the Startup Screen.

See our **Installing Studuino Software** manual for details on how to open the environment on Mac OS X and Raspberry Pi.

The Windows version of the Studuino Programming Environment opens with the Startup Screen shown here. Turn to the next page for an overview of this screen.

■ **The Programming Environments**

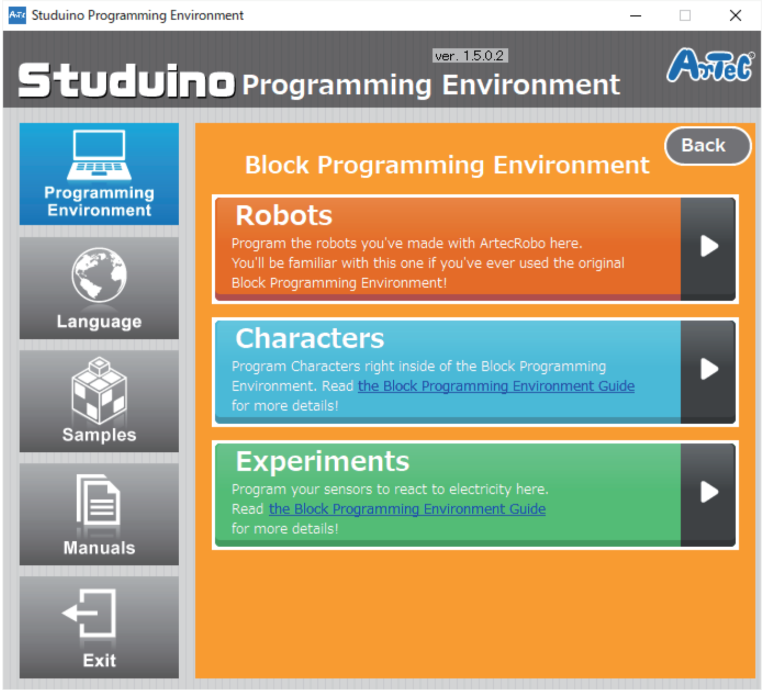
You'll see a panel that allows you to choose the software you'll use to program.

- **The Icon Programming Environment**

  Choose the **Icon Programming Environment** to open the software and begin programming your robot. Go to **5. The Studuino Icon Programming Environment** to learn how to program your robot using the Icon Programming Environment.

- **The Block Programming Environment**

  Choose the **Block Programming Environment** and you'll be able to choose from three different versions of the software (Robots, Characters, and Experiments).



Program the robots you've made with ArtecRobo here. You'll be familiar with this one if you've ever used the original Block Programming Environment!

Program Characters right inside of the Block Programming Environment. Read the Block Programming Environment Guide for more details!

Program your sensors to react to electricity here. Read the Block Programming Environment Guide for more details!

Choose the version that's most suitable for you.

Version	Purpose
Robots	Program your ArtecRobo robot. (See 6.2 for details)
Characters	Make games as well as animations. (See 6.3 for details)
Experiments	Control electricity using sensors. (See 6.4 for details)

■ **Language Settings**

Select **Language Settings** from the main menu to change the software language to English, French, Japanese (hiragana or kanji), Chinese (simplified or traditional), or Korean.

■ **Samples**

This section provides samples and instruction manuals of robots made with Artec Blocks, Studuino, and other parts.

■ **Manuals**

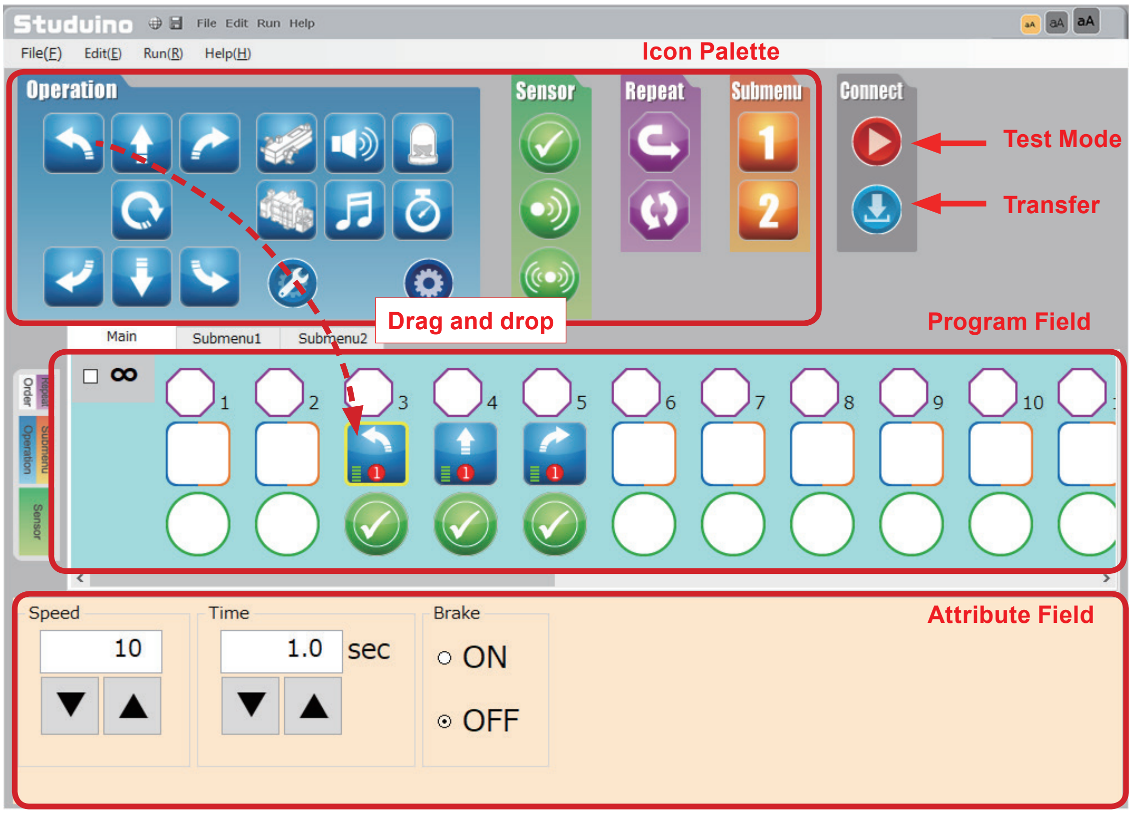
Lists manuals for the Studuino Programming Environment.

■ **Exit**

Close the Startup Screen and exit the software.

5. The Studuino Icon Programming Environment

5.1. Overview and Features



Speed: 10 ▼ ▲

Time: 1.0 sec ▼ ▲

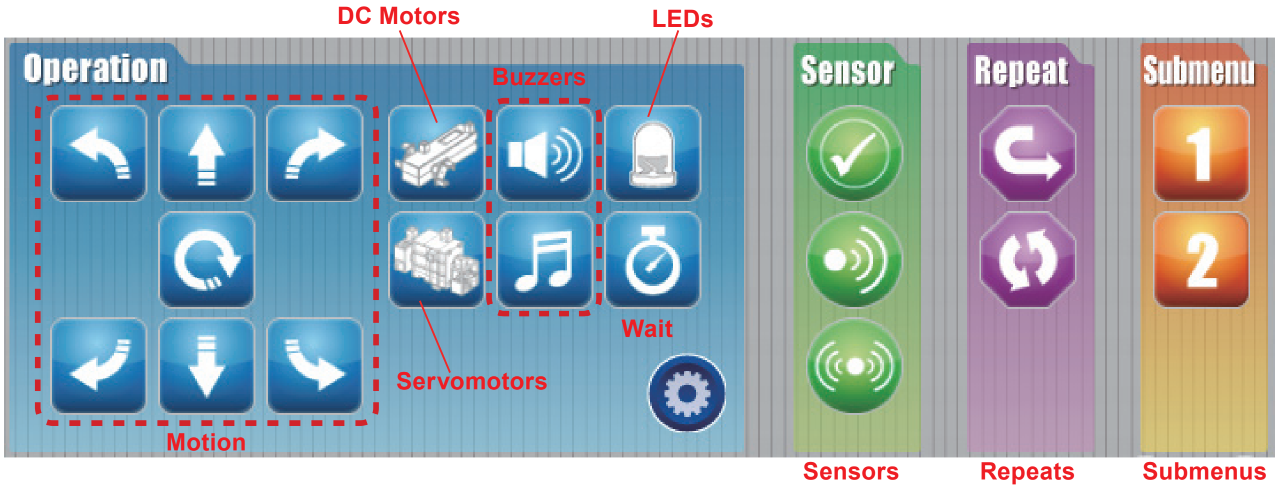
Brake: ○ ON ⊙ OFF

The Studuino Icon Programming Environment allows you to program your Artec robot by dragging and dropping icons into the Program Field.

Each icon in the Program Field can have a detailed mode of operation specified in its Attribute Field. You can easily create a full-fledged program for your robot by using Test Mode (see the section on **Test Mode** in **5.5 The Main Menu**) to control your robot in real time, or the Sensor Viewer (see the section on the **Sensor Viewer** in **5.5 The Main Menu**) to check the values of your sensors as you program.

Once you've created your program, you can use the ⬇ Transfer button to transfer and run it on your Studuino. You can also convert your program to Arduino language and use the Arduino IDE to edit it (see the section on the **Show Arduino Language** feature in **5.5 The Main Menu**).

5.2. The Icon Palette



The icons you'll need to control your robot are found in the Icon Palette. The Operation Group contains icons that control your robot, the Sensor Group contains icons that control sensor values, the Repeat Group contains icons that control program loops, and the Submenu Group contains icons that run Submenu items. The following section explains the different types of icons in each group.

■ **The Operation Group**

The Operation Group contains the icons you'll use to control the parts of your robot.

Controls DC Motors

Controls Servomotors

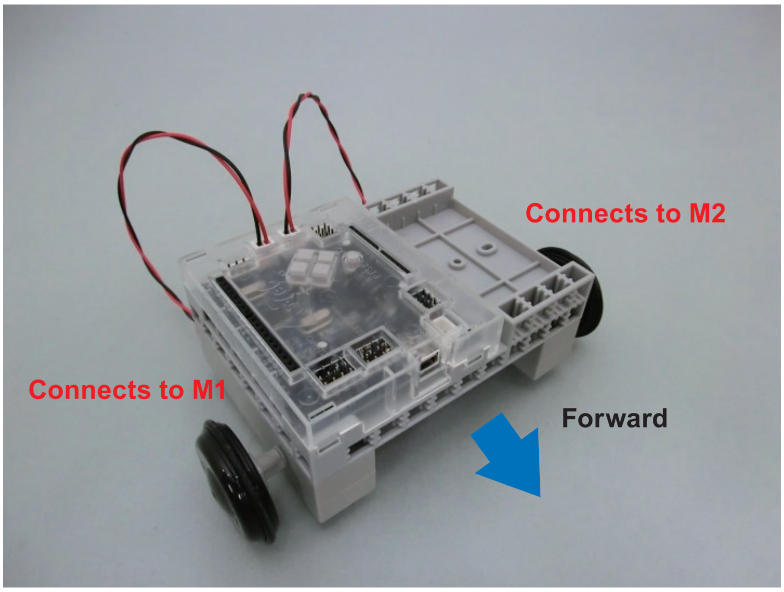
Controls LEDs

Plays a melody from a Buzzer

Plays a single note from the Buzzer

Waits for a specified time

As shown below, when two DC Motors are connected to M1 and M2 as car wheels you can use these icons to control the movement of your car. The wheels should be in front.



Forward	Backward
Left turn (F)	Left turn (B)
Right turn (F)	Right turn (B)
Rotate	

■ The Sensor Group

In the Sensor Group you'll find icons that use sensor values as conditions for your robot's movements.

Don't use sensor values as conditions for robot movements

Use one sensor value as a condition for robot movements

Use two sensor values as conditions for robot movements

■ The Repeat Group

The Repeat Group contains icons that are used to loop a part of your program.

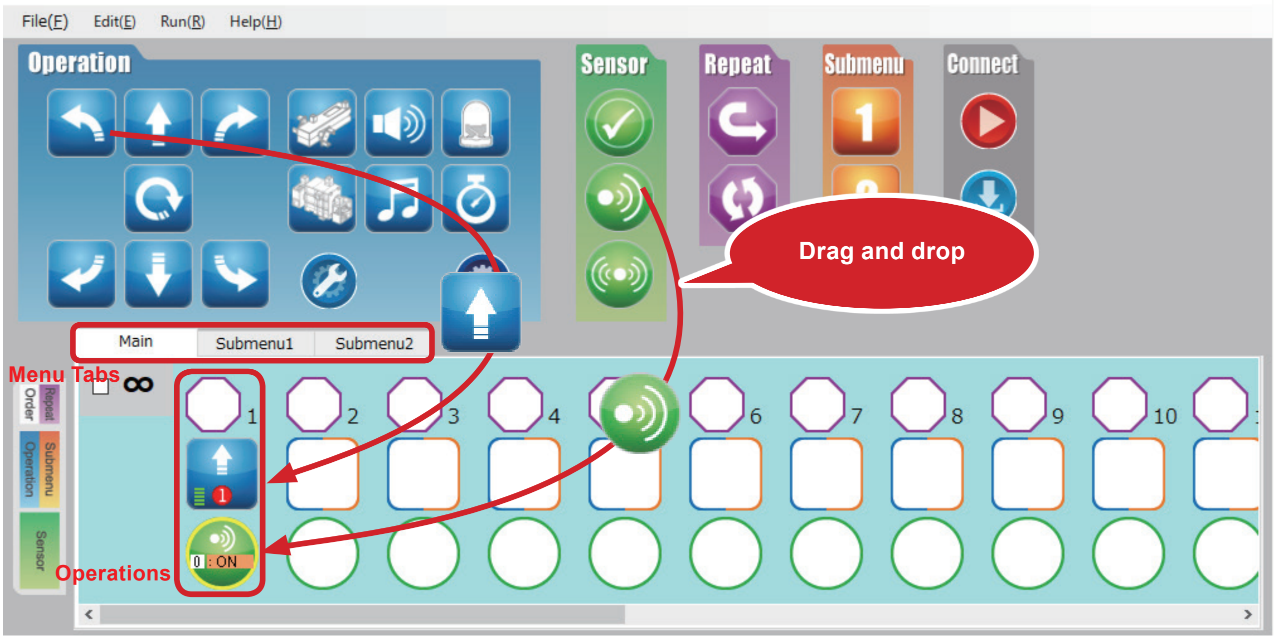
Start Repeat	End Repeat

■ The Submenu Group

The Submenu Group contains icons that run a specific operation in a Submenu.

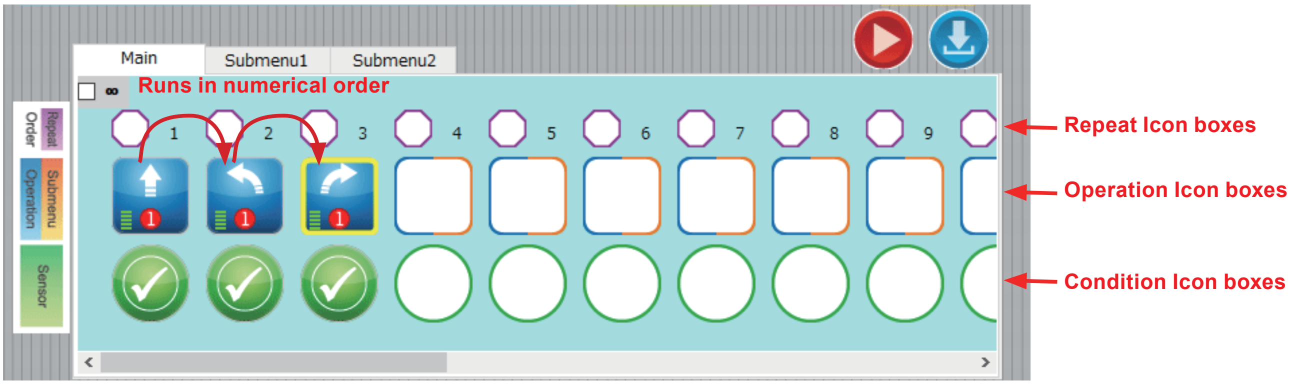
Run Submenu 1	Run Submenu 2

5.3. The Program Field



You can use the Program Field to program your robot by dragging and dropping control icons from the Icon Palette. Each column is treated as one operation. Use the tabs at the top to switch between the Main Menu, Submenu 1, and Submenu 2.

- Menu

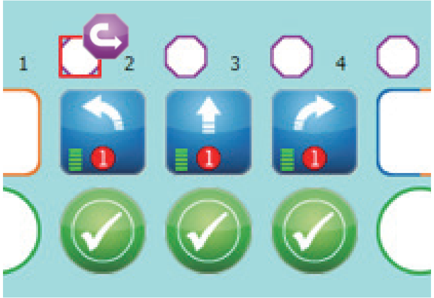


The menu is used to create programs for your robot. You can program up to 24 operations. Any program you create is run in numerical order. The menu, from top to bottom, contains boxes for Repeat, Operation, and Condition Icons. Place Repeat Icons in Repeat boxes, Operation or Submenu Icons in Operation boxes, and Sensor Icons in Condition boxes.

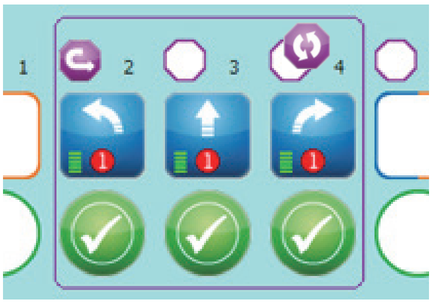
■ **Repeat Boxes**

Repeat Icons can be dragged and dropped into Repeat boxes. Follow the steps below to set Repeat Icons.

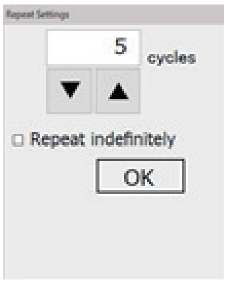
① Drop a **Start Repeat** Icon in the Repeat box of the Operation which you want to repeat.



② Drop an **End Repeat** Icon in the last Repeat box of the Operation which you want to repeat. You'll see a box appear around the section of your program that repeats.



③ After dropping ② , a **Repeat Settings** dialog box will appear where you can enter the number of times you want the Operation to repeat. If you check **Repeat indefinitely**, then it will repeat forever.


Repeat Settings

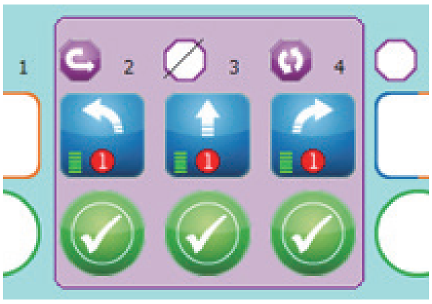
5 cycles

▼ ▲

□ Repeat indefinitely

OK

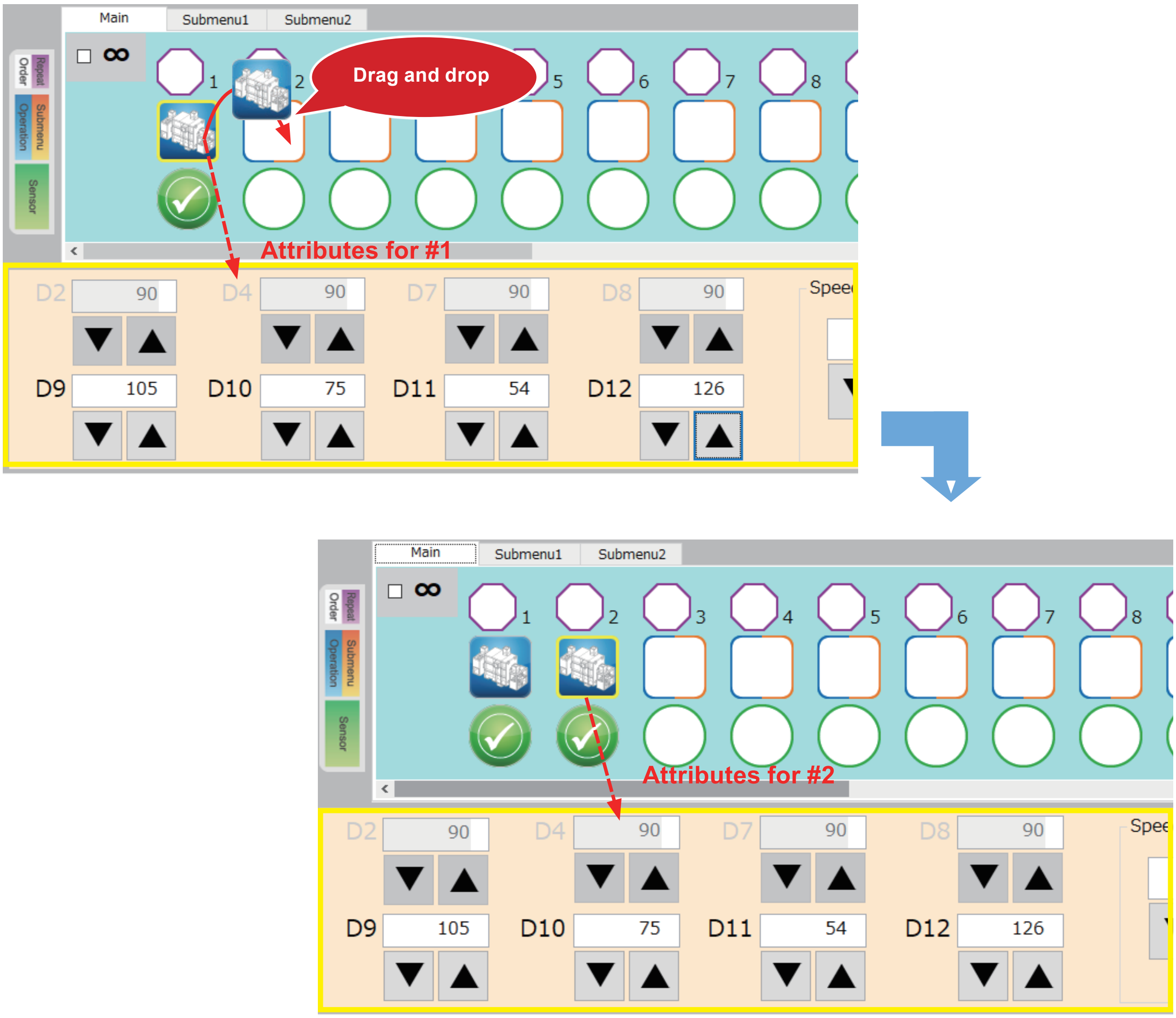
④ You have now made a Repeat. To change the number of Repeats, click the End Repeat Icon to open the Repeat Settings dialog box.



■ **Operation Boxes**

Operation and Submenu Icons can be dragged and dropped into Operation boxes. Submenu Icons 1 and 2 are used to run Operations you've placed in Submenu 1 and 2 of the Program Field (see the Submenus section for details).

You can copy any icon in an Operation box by dragging and dropping it into an empty box. As shown below, any icon you copy by dragging and dropping will keep the attributes of its parent icon.



D2 90 D4 90 D7 90 D8 90

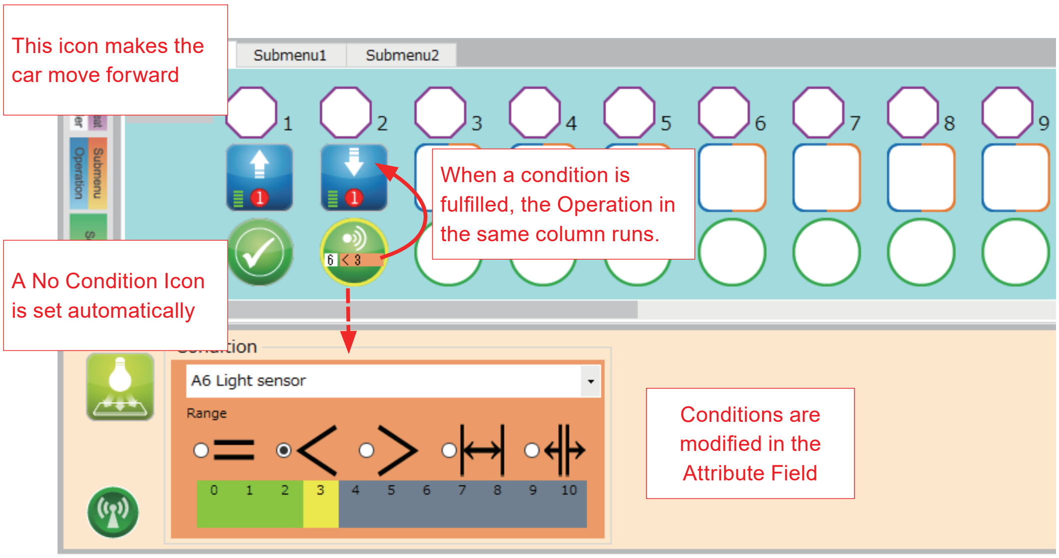
D9 105 D10 75 D11 54 D12 126

D2 90 D4 90 D7 90 D8 90

D9 105 D10 75 D11 54 D12 126

Copied icons keep the attributes of the parent icon

Condition boxes



This icon makes the car move forward

A No Condition Icon is set automatically

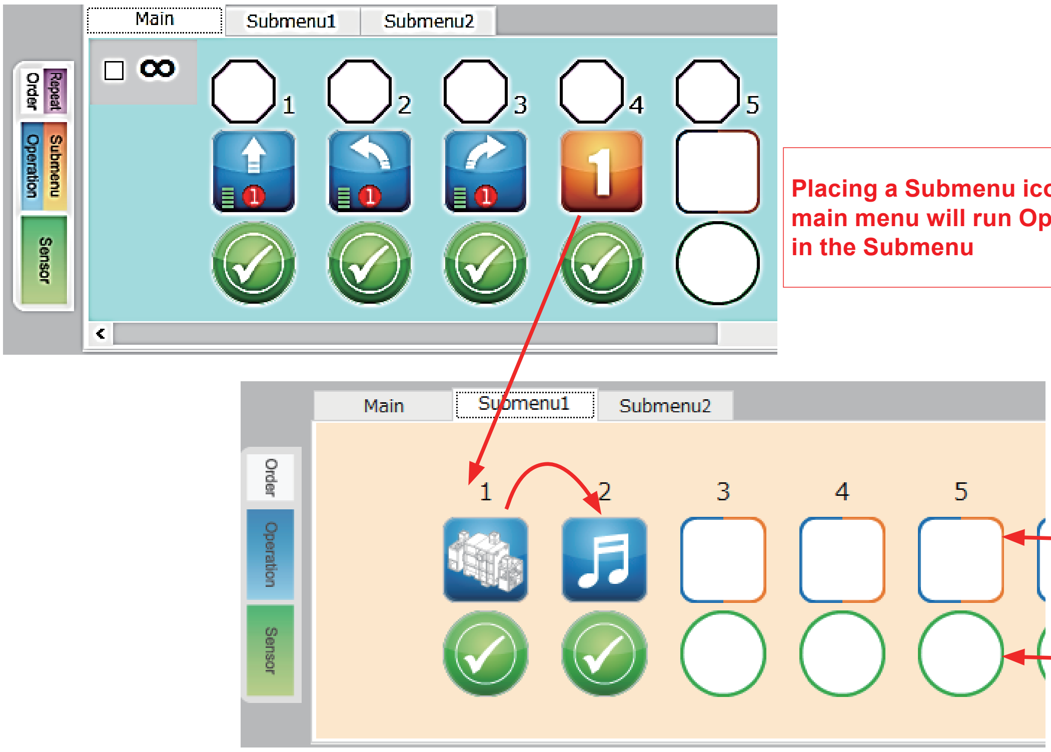
When a condition is fulfilled, the Operation in the same column runs.

Conditions are modified in the Attribute Field

Use Condition boxes to set conditions that modify the icon located in the same column. Dropping an icon in an Operation box will automatically make a ✓ No Condition Icon appear in the Condition box. Dropping Sensor Icons into Condition boxes allows you to use sensor values as a condition for an Operation. Sensor or threshold values which are used for the condition can be set in the Attribute Field. Selecting a Sensor Icon with a condition allows you to modify that condition in the Attribute Field. See **5.4. The Attribute Field** for details on how to use Sensor Icons to modify conditions.

The above picture shows a program which has an unconditional Operation that makes the car move forward in box 1, but a conditional Operation in step 2 that makes the car move backwards if the value of the Light Sensor is less than 3. This Operation will not run if the value of the Light Sensor is greater than 3.

Submenus



Placing a Submenu icon in the main menu will run Operations in the Submenu

Condition Icon boxes

Operation Icon boxes

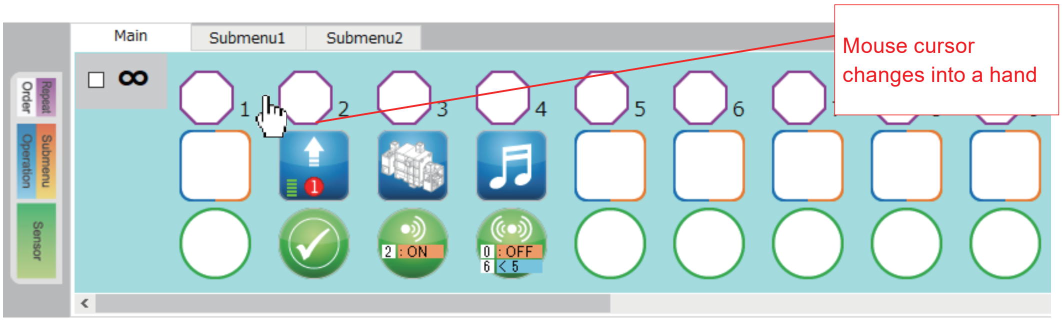
Smaller programs with up to eight Operations can be created in Submenus. When a Submenu Icon is placed in the Main Menu, the program created in the Submenu is run in numerical order. There are Operation and Condition boxes in Submenus. Operation Icons are placed in Operation boxes and Sensor Icons in Condition boxes.

Selecting and Placing Multiple Icons

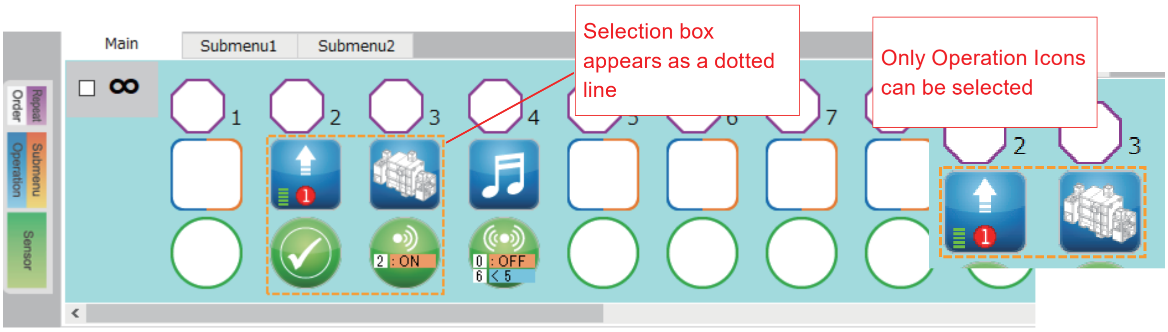
Multiple icons in the Program Field can be copied by selecting, then dragging and dropping them into another box.

Click on any space outside the Operation, Condition and Repeat boxes in the Program Field and the mouse cursor will change to ☝. Drag your cursor to create a selection box around the icons you want to copy. Release the left mouse button once you have selected the icons and your cursor will return to normal.

★ Only Operation Icons can be selected. Empty boxes and Submenu Icons will not be included in your selection.

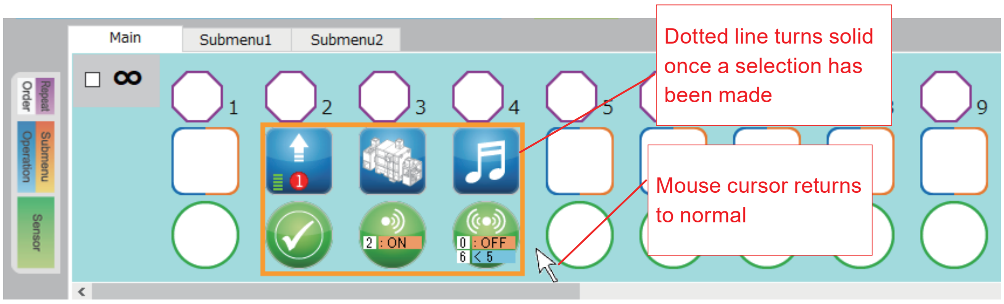


Mouse cursor changes into a hand



Selection box appears as a dotted line

Only Operation Icons can be selected

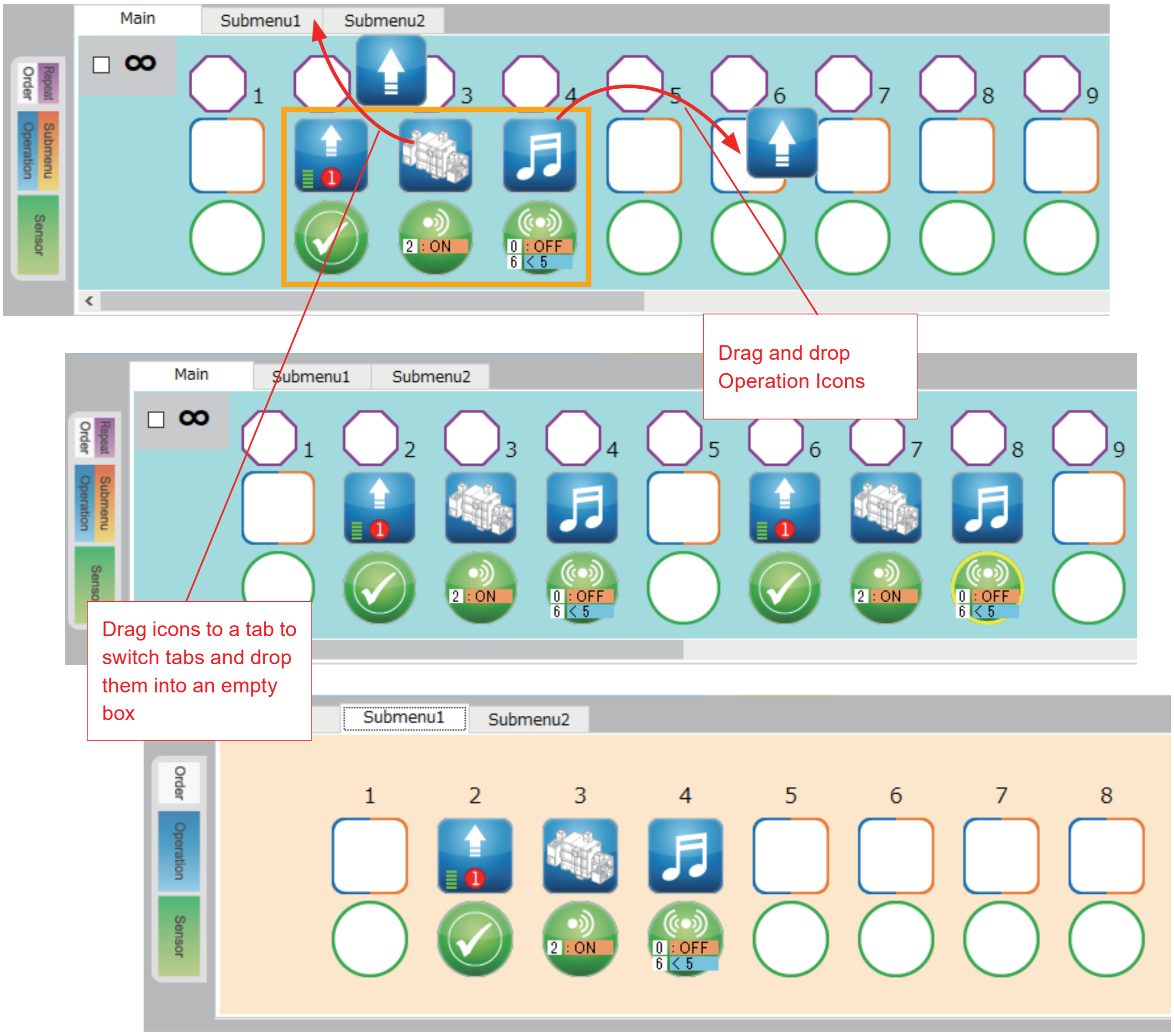


Dotted line turns solid once a selection has been made

Mouse cursor returns to normal

Now drag the icons to a new space. Your mouse cursor will turn into the first icon in the selection. Dragging the icons to another space will copy them. Dragging your cursor to a different menu tab will switch to that menu. Dragging the icons to boxes in a different menu tab will copy them there.

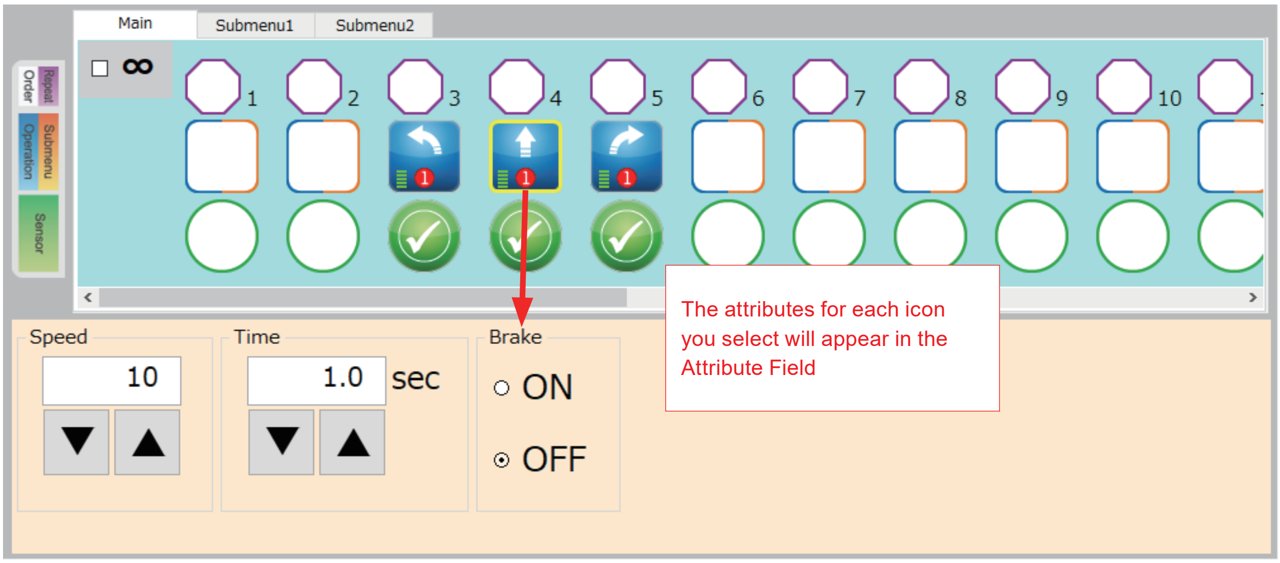
★ Copying Operation Icons will also copy any Condition Icons that you have set for them. Dragging Operation Icons with no condition will set a ✓ No Condition Icon for them.



Drag and drop Operation Icons

Drag icons to a tab to switch tabs and drop them into an empty box

5.4. The Attribute Field



Speed 10 ▼ ▲

Time 1.0 sec ▼ ▲

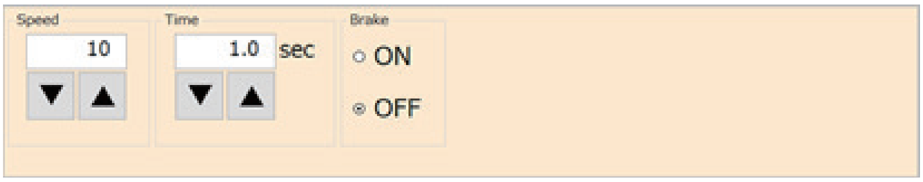
Brake ○ ON ⊙ OFF

The attributes for each icon you select will appear in the Attribute Field

Each Operation Icon has unique attributes. For example, a ⬆ Motion Icon for a car will show the speed and duration of its movements. This information can be modified in the Attribute Field. Select any icon in the Program Field to show and change its attributes in the Attribute Field.

The following section describes attributes for different icons.

- **Motion Icon 1**



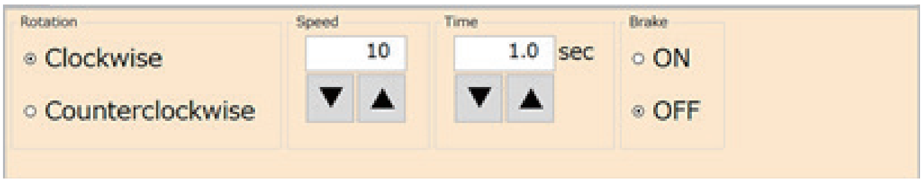
Speed 10 ▼ ▲

Time 1.0 sec ▼ ▲

Brake ○ ON ⊙ OFF

Attribute	Contents
Speed	Set speed from 0-10.
Time	Set time from 0.1 to 25.5 sec.
Brake	Stop your robot. This icon is useful for when you need to make your robot stop at a certain place. Choose OFF and your robot will coast to a stop once it finishes its program. This will also give your robot a smoother transition between different motions.

Motion Icon 2



Rotation: ● Clockwise ○ Counterclockwise

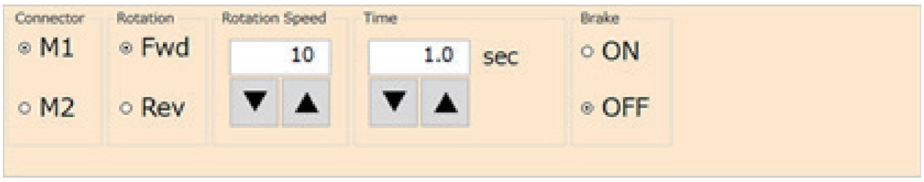
Speed: 10 ▼ ▲

Time: 1.0 sec ▼ ▲

Brake: ○ ON ● OFF

Attribute	Contents
Speed	Set speed from 0-10.
Rotation	Turn clockwise or counterclockwise.
Time	Set a time from 0.1 to 25.5 sec.
Brake	Stop your robot. This icon is useful for when you need to make your robot stop at a certain place. Choose OFF and your robot will coast to a stop once it finishes its program. This will also give your robot a smoother transition between different motions.

DC Motor Icon



Connector: ● M1 ○ M2

Rotation: ● Fwd ○ Rev

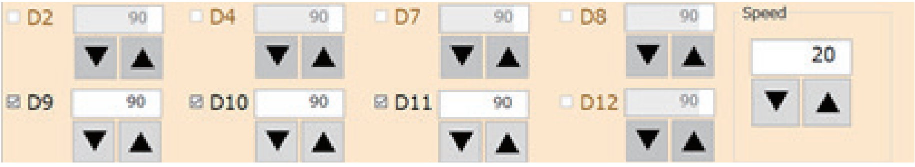
Rotation Speed: 10 ▼ ▲

Time: 1.0 sec ▼ ▲

Brake: ○ ON ● OFF

Attribute	Contents
Connector	Sets the connector for the corresponding DC Motor. See **Port Settings** in **5.5. The Main Menu** for details on which connectors you can use.
Rotation	Sets the rotation direction (forward or reverse).
Speed	Sets the speed of the DC Motor from 0-10.
Time	Set movement time from 0.1 to 25.5 sec.
Brake	Stop your DC Motor. This icon is useful for when you need to make your robot stop at a certain place. Choose OFF and your robot will coast to a stop once it finishes its program. This will also give your robot a smoother transition between different motions.

Servomotor Icon



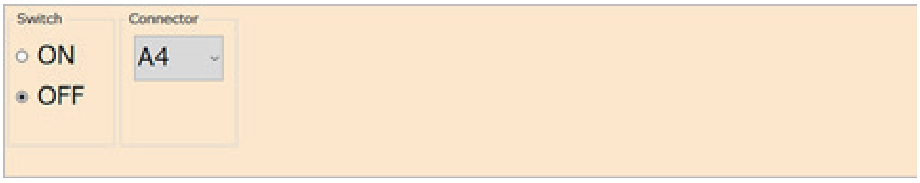
□ D2 90 ▼ ▲ □ D4 90 ▼ ▲ □ D7 90 ▼ ▲ □ D8 90 ▼ ▲

☑ D9 90 ▼ ▲ ☑ D10 90 ▼ ▲ ☑ D11 90 ▼ ▲ □ D12 90 ▼ ▲

Speed: 20 ▼ ▲

Attribute	Contents
Angle	Adjust the number in the box for each motor to set an angle from 0 to 180 degrees in one degree increments. **See Port Settings in 5.5**. The Main Menu for details on which connectors you can use.
Speed	Sets 20 levels of speed for the selected angles.

■ LED Icon



Switch

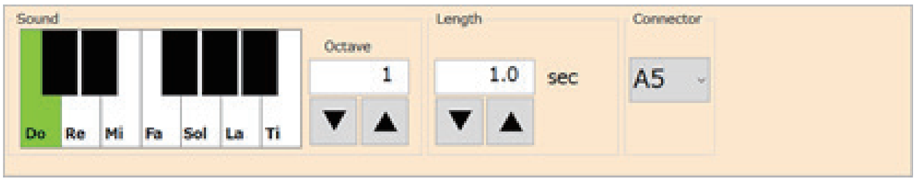
○ ON

● OFF

Connector: A4

Attribute	Contents
Switch	Turns the LED on or off.
Connector	Choose the connector for the corresponding LED. See Port Settings in **5.5. The Main Menu** for details on which connectors you can use.

■ Buzzer Icon



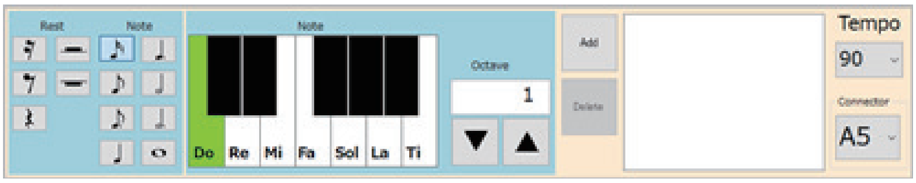
Octave: 1

Length: 1.0 sec

Connector: A5

Attribute	Contents
Sound	Sets a note to play with a length of 0.1 to 1.0 seconds.
Connector	Choose the connector for the corresponding Buzzer. See **Port Settings** in **5.5. The Main Menu** for details on which connectors you can use.

■ Melody Icon



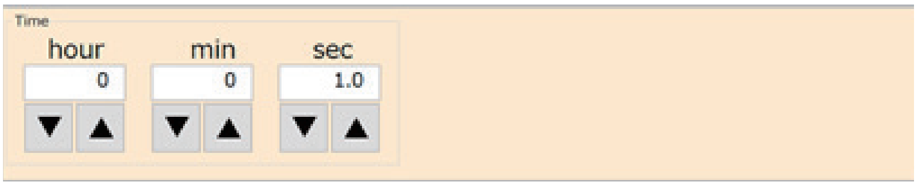
Octave: 1

Tempo: 90

Connector: A5

Attribute	Contents
Compose	Add notes by scale or rests by selecting them and pressing the → key. Choose from a tempo of either 90, 120, or 150. Each icon can hold eight notes and/or rests.
Connector	Choose the connector for the corresponding Buzzer. See **Port Settings** in **5.5. The Main Menu** for details on which connectors you can use.

■ **Wait Icon**



Time

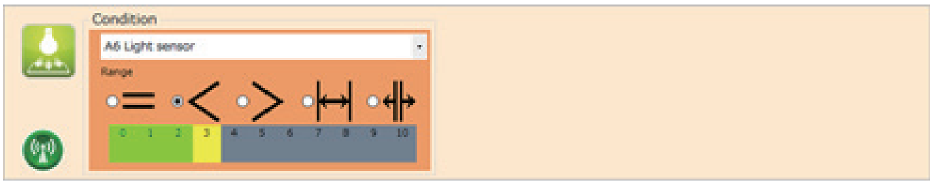
hour 0

min 0

sec 1.0

Attribute	Contents
Time	Waits for length of time specified in hours, minutes, and seconds.

■ **Single Condition Icon**

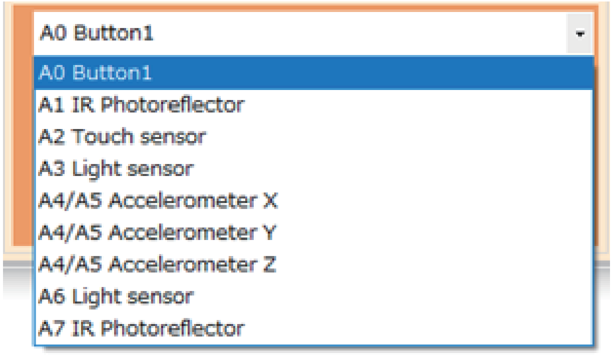


Condition

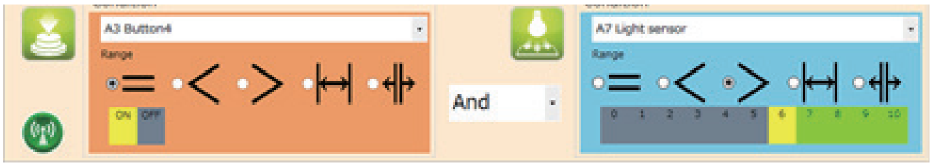
A6 Light sensor

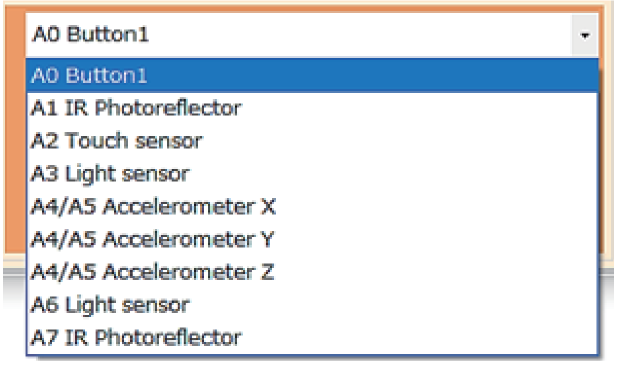
Range

○= ●< ○> ○|↔| ○↔||↔

Attribute	Contents
Condition	Set a condition threshold by using the combo box to choose a sensor connected to the Studuino and selecting the type of condition (equality (=), inequality (< >), or range (|→←|, ←||→)). Ranges are open intervals and do not include the threshold numbers. The picture below shows the combo box and the sensors connected to the Studuino. See **Port Settings** in **5.5. The Main Menu** for details on how the sensors will be displayed.  A0 Button1 A0 Button1 A1 IR Photoreflector A2 Touch sensor A3 Light sensor A4/A5 Accelerometer X A4/A5 Accelerometer Y A4/A5 Accelerometer Z A6 Light sensor A7 IR Photoreflector

■ Dual Condition Icon

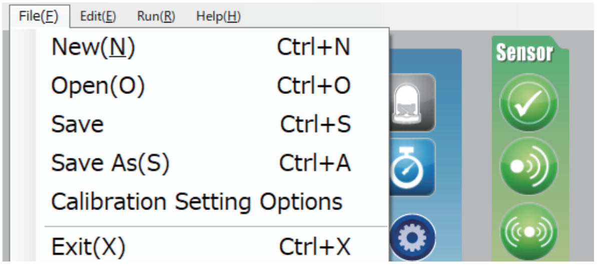


Attribute	Contents
Condition	Set a condition threshold by using the combo box to choose a sensor connected to the Studuino and selecting the type of condition (equality (=), inequality (< >), or range (|→←|, ←||→)). Ranges are open intervals and do not include the threshold numbers. A Dual Condition Icon has an extra combo box for choosing an if/else statement. The picture below shows the combo box and the sensors connected to the Studuino. See **Port Settings** in **5.5. The Main Menu** for details on how the sensors will be displayed.  A0 Button1 A0 Button1 A1 IR Photoreflector A2 Touch sensor A3 Light sensor A4/A5 Accelerometer X A4/A5 Accelerometer Y A4/A5 Accelerometer Z A6 Light sensor A7 IR Photoreflector

5.5. The Main Menu

■ **The File Menu**

You can use the File menu to save and load files.

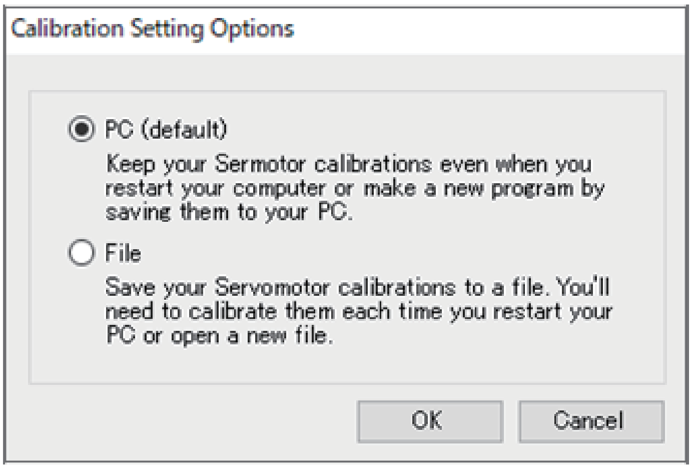


- **Open**
  Open a previously saved file.

- **Save**
  Saves the program you're currently working on and overwrites the previous version.

- **Save As**
  Saves the program with a specified name.

- **Calibration Setting Options**
  Use this setting dialog to choose whether to save your Motor Calibration settings directly to your PC or to a file.



Calibration Setting Options

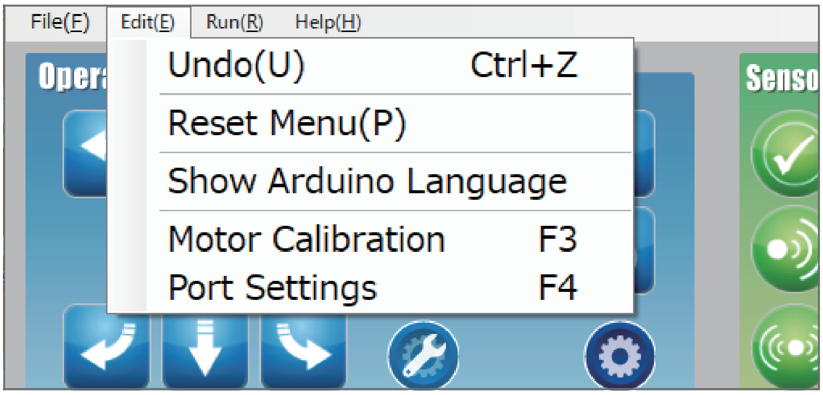
● PC (default)
Keep your Sermotor calibrations even when you restart your computer or make a new program by saving them to your PC.

○ File
Save your Servomotor calibrations to a file. You'll need to calibrate them each time you restart your PC or open a new file.

- **Exit**
  Closes the Studuino Icon Programming Environment.

■ The Edit Menu

The Edit menu allows you to make or edit programs. Only version 2.0 of the Icon Programming Environment allows you to use optional parts or register IR signals.



- **Undo**
  Undoes your last action.
  The software will remember the last 10 actions you've performed, including placing icons, inserting and deleting columns, and changing Port Settings.

- **Reset**
  Removes all icons from the Program Field.

- **Show Arduino Language**
  Converts the program in your Program Field to Arduino language. The source code you export using this feature can be compiled and sent to your Studuino using Arduino IDE.

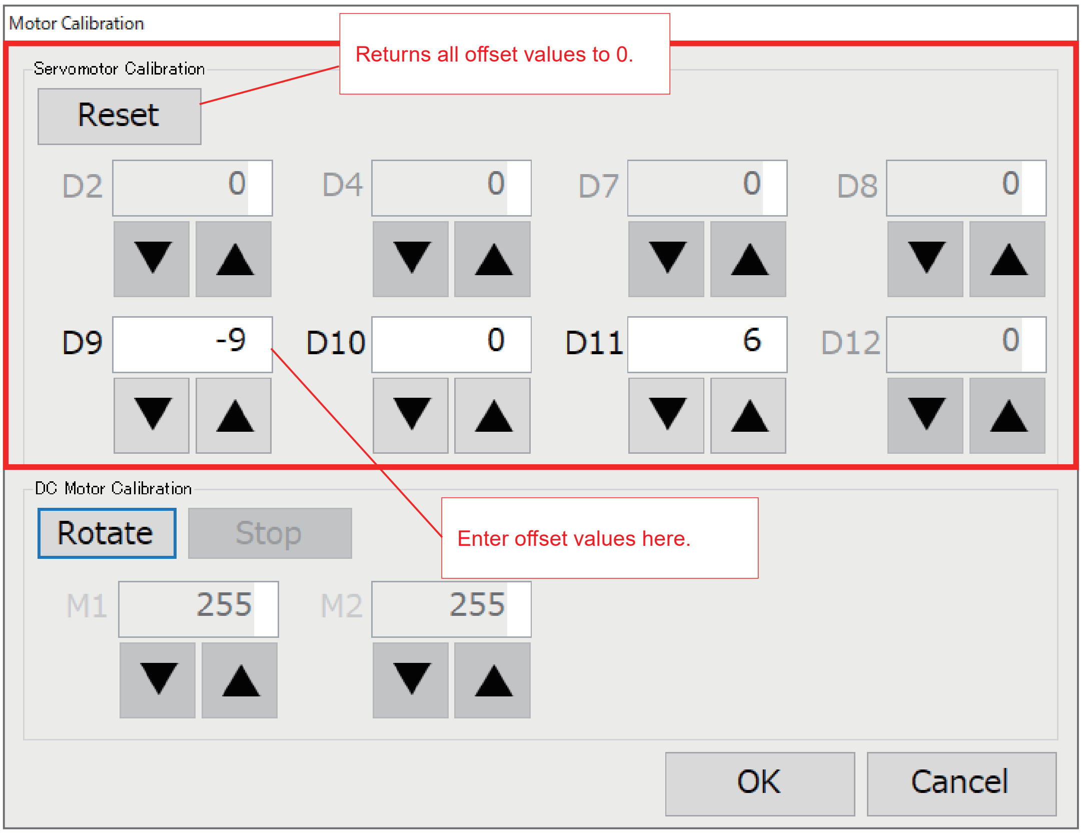
- **Optional Parts (IPE 2.0 only)**
  Check this option if you wish to use Gyroscopes, Color Sensors, Ultrasonic Sensors, Temperature Sensors, and Bluetooth Modules. Visit our website and download the **Optional Parts Guide** for details on how to use these parts.

- **Register IR Signals (IPE 2.0 only)**
  This opens the registration window, allowing you to register IR signals. You'll need to enable optional parts and enable your IR Receiver in Port Settings.

- **Motor Calibration**
  Use this feature to adjust the angles of your Servomotors or the speed of your DC Motors and save the settings to a file. Opening this feature will start Test Mode and open the Motor Calibration dialog box. Click the 🔧 Motor Calibration Icon under Operation to switch to Test Mode and open the Motor Calibration dialog.

- **Servomotor Calibration**

When the dialog box opens, all angles for the connected Servomotors are set to 90 degrees by default. Entering an offset value from -15 to 15 degrees will set the angles for the corresponding Servomotor to 90 degrees + the offset value.



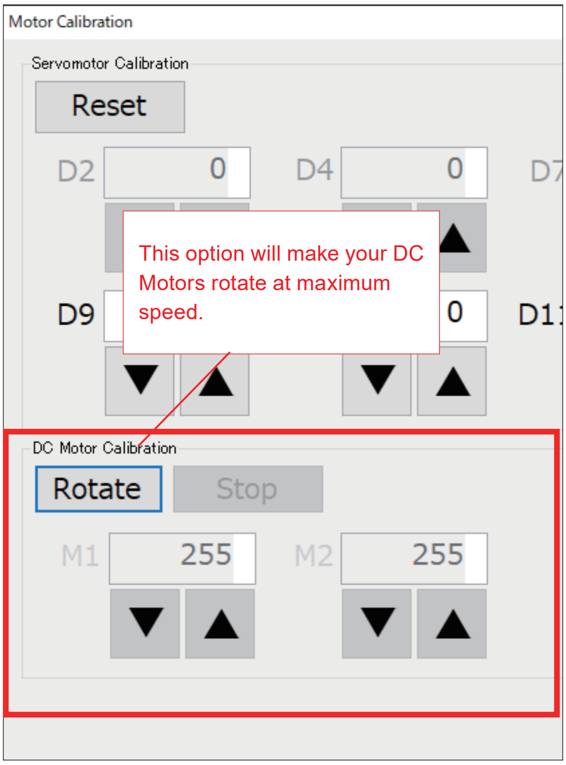
Returns all offset values to 0.

Enter offset values here.

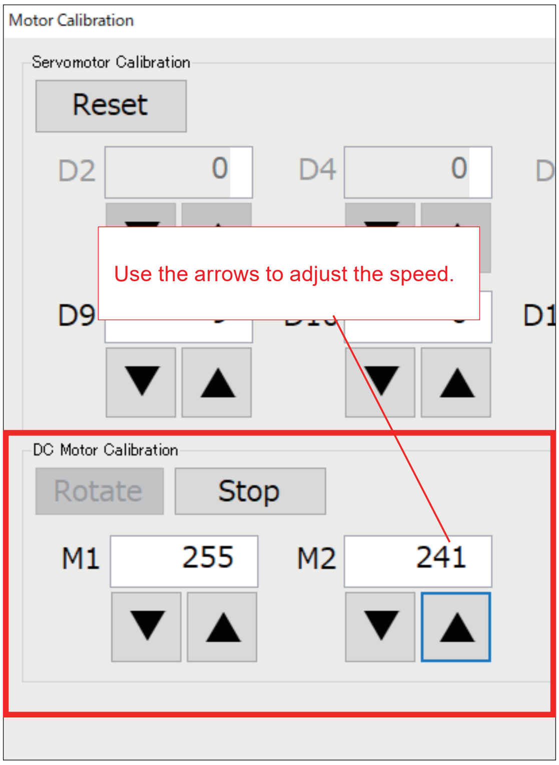
Watch your Servomotor and adjust the values until you get the correct 90 degree angle.

Click the **OK** button once you've finished calibrating your motors. The settings will be saved and the same values will be used every time you start the software.

- **DC Motor Calibration**



This option will make your DC Motors rotate at maximum speed.

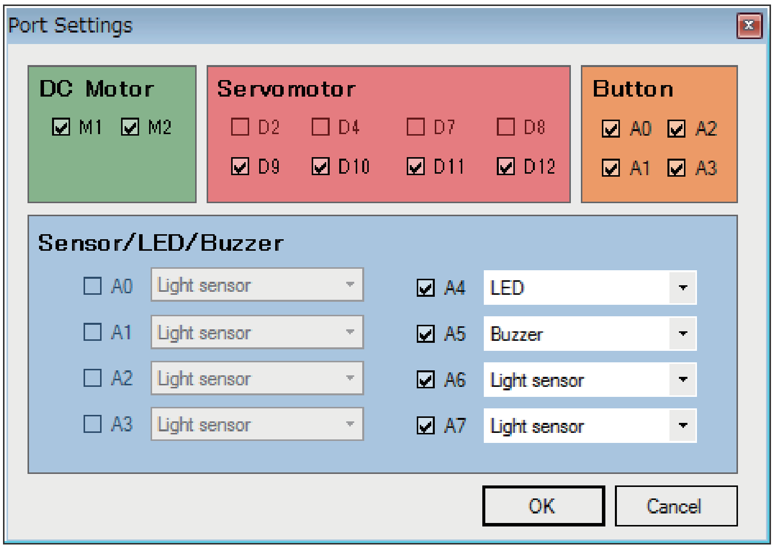


Use the arrows to adjust the speed.

Once the dialog opens, click the **Rotate** button to make your DC Motors rotate at maximum speed. Doing this allows you to click the arrows until your motors are rotating at identical speeds. Click the **Stop** button to stop your motors. This will enable the **OK** and **Cancel** buttons. Click **OK**. The settings will be saved and the same values will be used every time you start the software.

- **Port Settings**

Port Settings allows you to set the parts connected to your Studuino in the Icon Programming Environment. Open the Port Settings dialog by clicking ⚙ under the Operation Icons in the Icon Palette.



Port Settings

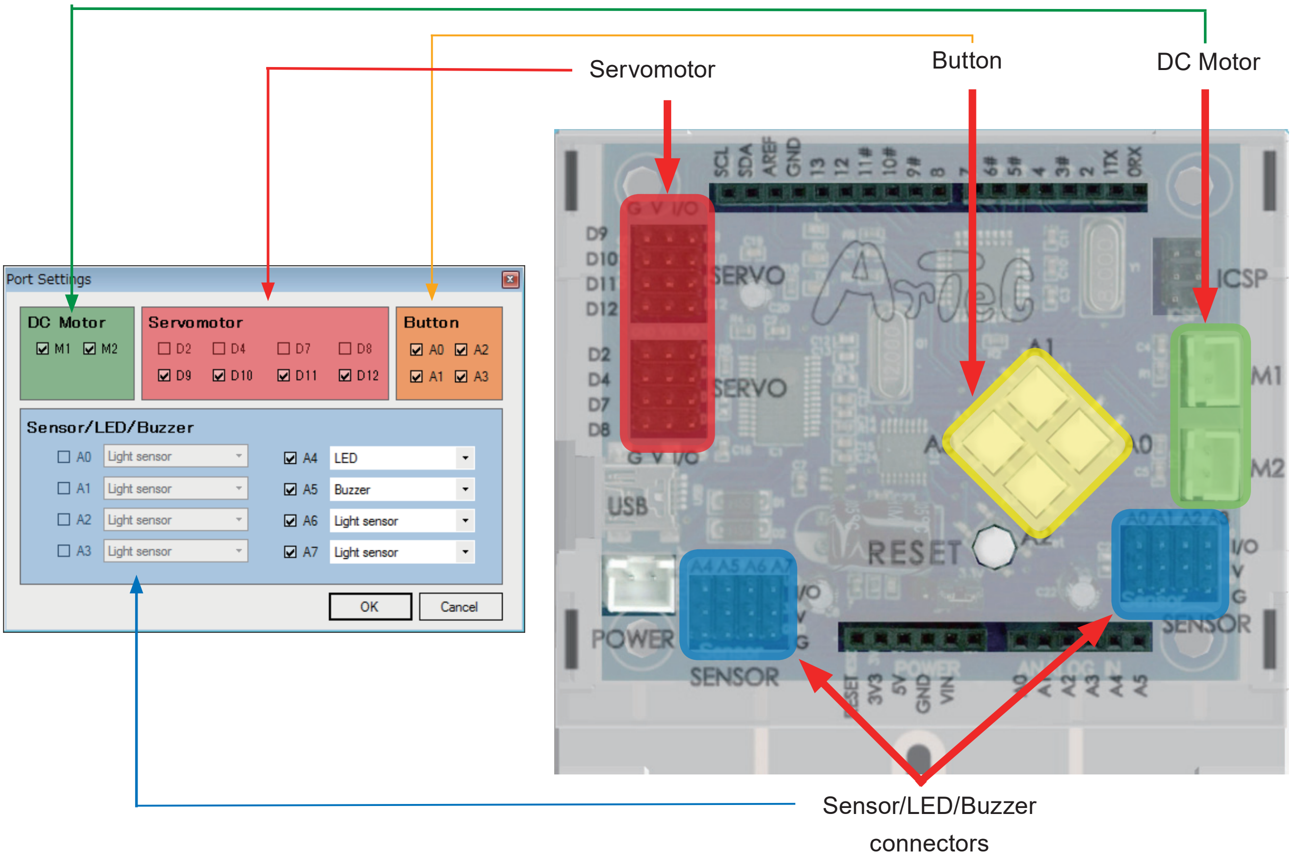
DC Motor
☑ M1 ☑ M2

Servomotor
☐ D2 ☐ D4 ☐ D7 ☐ D8
☑ D9 ☑ D10 ☑ D11 ☑ D12

Button
☑ A0 ☑ A2
☑ A1 ☑ A3

Sensor/LED/Buzzer
☐ A0 Light sensor
☐ A1 Light sensor
☐ A2 Light sensor
☐ A3 Light sensor
☑ A4 LED
☑ A5 Buzzer
☑ A6 Light sensor
☑ A7 Light sensor

As shown in the picture below, this dialog box shows all Studuino connectors and Push-buttons on the Studuino. Be sure to check any which have a part connected to them.



As described in **2.3. About Studuino**, DC Motor connector M1 and Servomotor connectors D2 and D4, DC Motor connector M2 and Servomotor connectors D7 and D8, Push-buttons A0-A3 and Sensor/LED/Buzzer connectors A0-A3 can't be used at the same time. As these combinations also can't be enabled at the same time in the Port Settings dialog, please uncheck one side while the other is active. For example, when using D2 and D4 for the Servomotor, the checkboxes for D2 and D4 would only be available when DC Motor box M1 is unchecked.

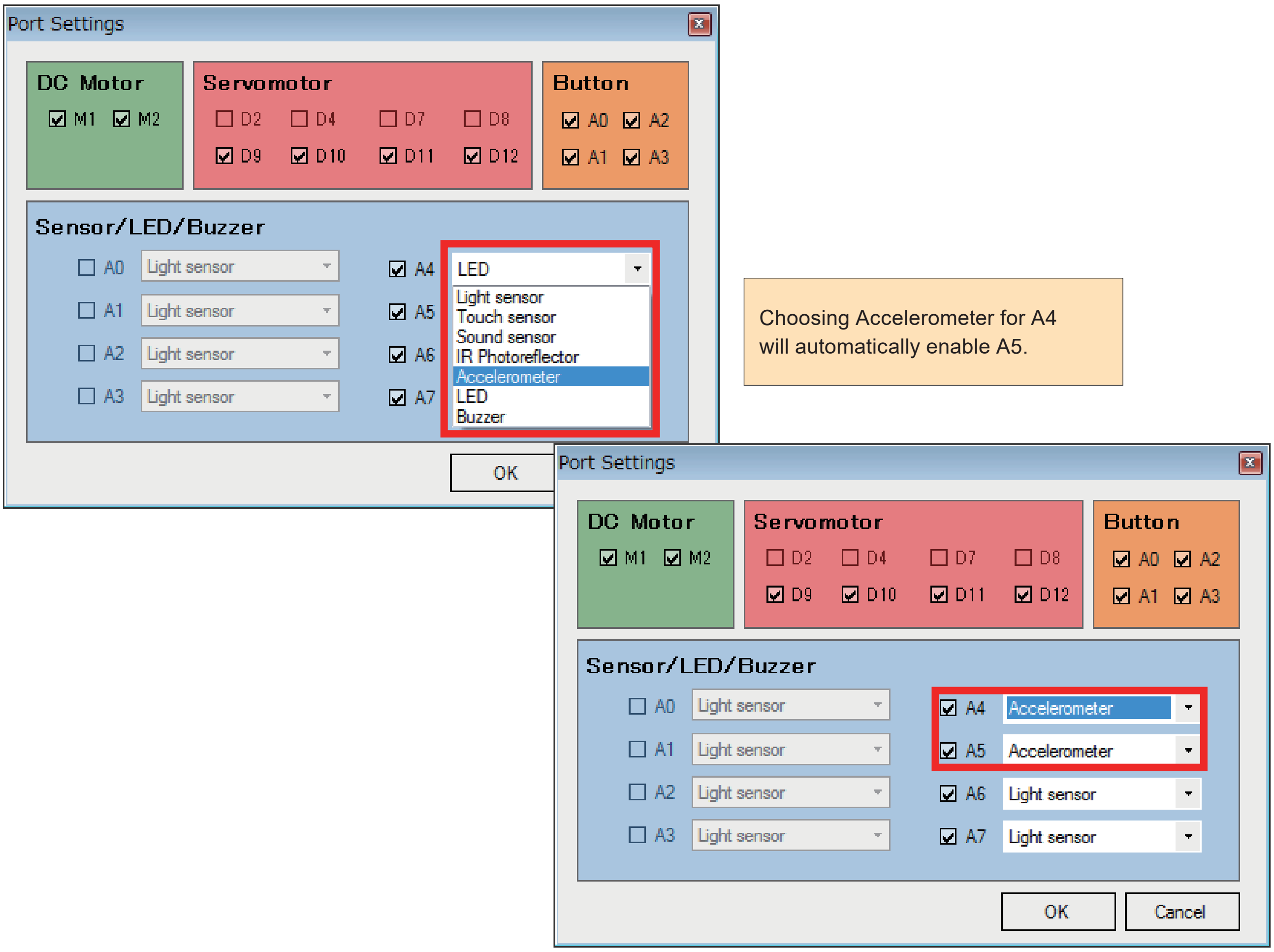
<Caution>

For an Accelerometer, which uses two connectors, all of the corresponding connectors should be checked.

Part	Connector Combination
Accelerometer	A4, A5

Connector combination when multiple connectors are used

When the checkboxes of the corresponding connectors in Connector Combination above are checked, the part connected to the Studuino will be displayed. Checking one box for a part which uses multiple connectors will automatically select the boxes for any other connectors it uses.



Port Settings

DC Motor: ☑ M1 ☑ M2

Servomotor: □ D2 □ D4 □ D7 □ D8 ☑ D9 ☑ D10 ☑ D11 ☑ D12

Button: ☑ A0 ☑ A2 ☑ A1 ☑ A3

Sensor/LED/Buzzer

□ A0 Light sensor
□ A1 Light sensor
□ A2 Light sensor
□ A3 Light sensor
☑ A4 LED
☑ A5
☑ A6
☑ A7

Choosing Accelerometer for A4 will automatically enable A5.

Port Settings

DC Motor: ☑ M1 ☑ M2

Servomotor: □ D2 □ D4 □ D7 □ D8 ☑ D9 ☑ D10 ☑ D11 ☑ D12

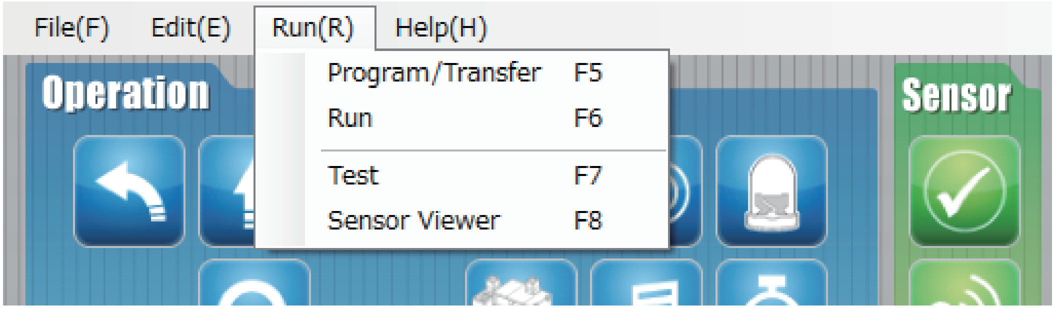
Button: ☑ A0 ☑ A2 ☑ A1 ☑ A3

Sensor/LED/Buzzer

□ A0 Light sensor
□ A1 Light sensor
□ A2 Light sensor
□ A3 Light sensor
☑ A4 Accelerometer
☑ A5 Accelerometer
☑ A6 Light sensor
☑ A7 Light sensor

■ **The Run Menu**

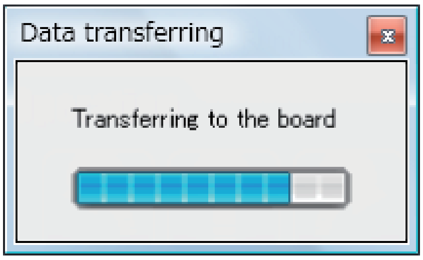
The Run menu allows you to connect to your Studuino while programming.



- **Transfer**

Transfer compiles any program you've made and sends it to your Studuino. You can also perform a transfer by clicking the ⬇ button between the Program Field and Icon Palette.

Make sure that your Studuino is connected to your PC via a USB cable before transferring your program. You'll see the dialog box below as the program transfers. The transfer has finished once it disappears.



Data transferring

Transferring to the board

- **Run**

Run executes any program you've transferred.

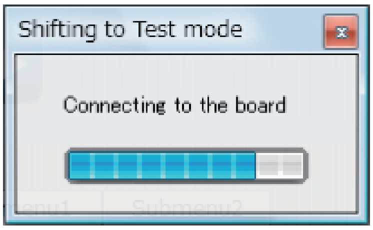
- **Test**

Click Test or the ▶ button between the Program Field and Icon Palette to start Test Mode. You can use Test Mode to communicate with your Studuino and control parts in real time. You can also use it to adjust icon settings in the Attribute Field to fine-tune your robot.

Read below to find out how each icon works in Test Mode.

Icon	Control
Motion	Click on any Motion Icon in the Program Field to see its movement settings.
DC Motor	Click on any DC Motor Icon in the Program Field to see its movement settings.
Servomotor	Click on any Servomotor Icon in the Program Field to see its angle settings. You can also check angle settings in real time using the Attribute Field.
LED	Changing LED Icon settings will turn them on and off.
Buzzers	The Buzzer will play notes that you set in the Attribute Field.
Melody	Click on any Melody Icon in the Program Field to hear the melody you've programmed. The Buzzer will play the notes as you change them using the Attribute Field.

Make sure your PC and Studuino are connected before starting Test Mode. You'll see the dialog box below and Test Mode will open once it disappears.

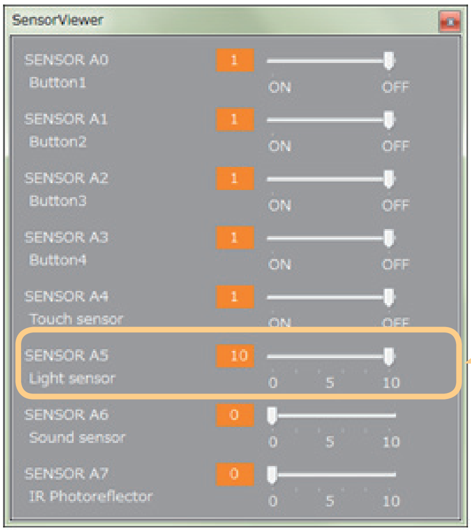


Shifting to Test mode

Connecting to the board

■ The Sensor Viewer

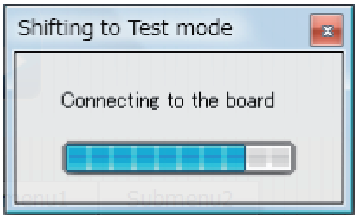
Click the button between the Program Field and Icon Group to open the Sensor Viewer. The Sensor Viewer is used to communicate with your Studuino and check the values of any sensor connected to the unit.



SENSOR A0 Button1	1	ON OFF
SENSOR A1 Button2	1	ON OFF
SENSOR A2 Button3	1	ON OFF
SENSOR A3 Button4	1	ON OFF
SENSOR A4 Touch sensor	1	ON OFF
SENSOR A5 Light sensor	10	0 5 10
SENSOR A6 Sound sensor	0	0 5 10
SENSOR A7 IR Photoreflector	0	0 5 10

You'll see the values for any sensor you've connected.

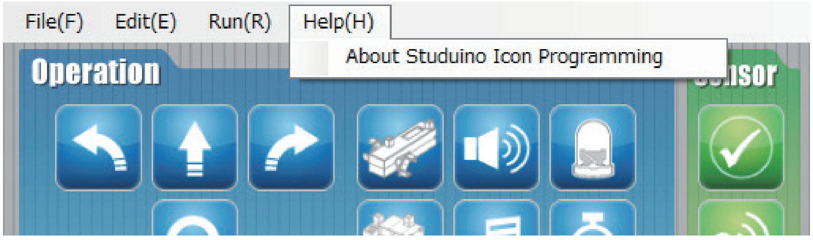
Make sure your PC and Studuino are connected before opening the Sensor Viewer. You'll see the dialog box below and the Sensor Viewer will open once it disappears.


Shifting to Test mode

Connecting to the board

■ Help Menu

The Help menu contains information about the Studuino Block Programming Environment.

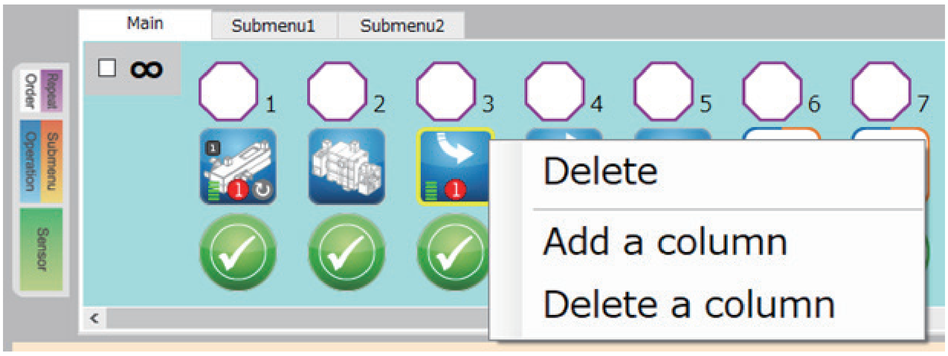


● About Studuino Icon Programming

Shows you information about the Icon Programming Environment.

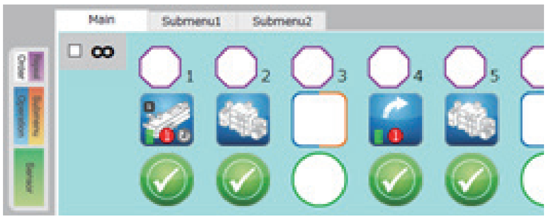
5.6. The Context Menu

Right-clicking on a box in the Program Field will bring up a context menu.



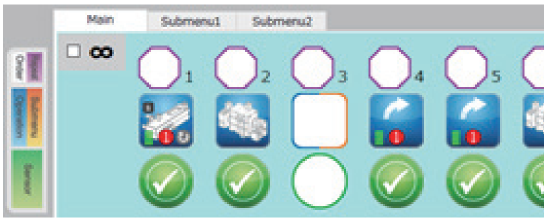
- **Delete**

Deletes an icon.



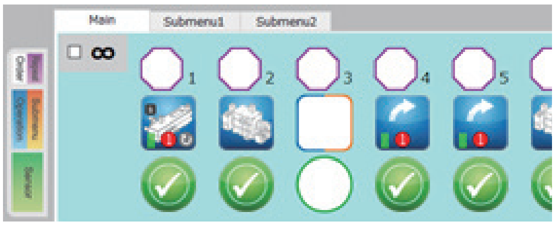
- **Add a column**

Inserts an empty column.



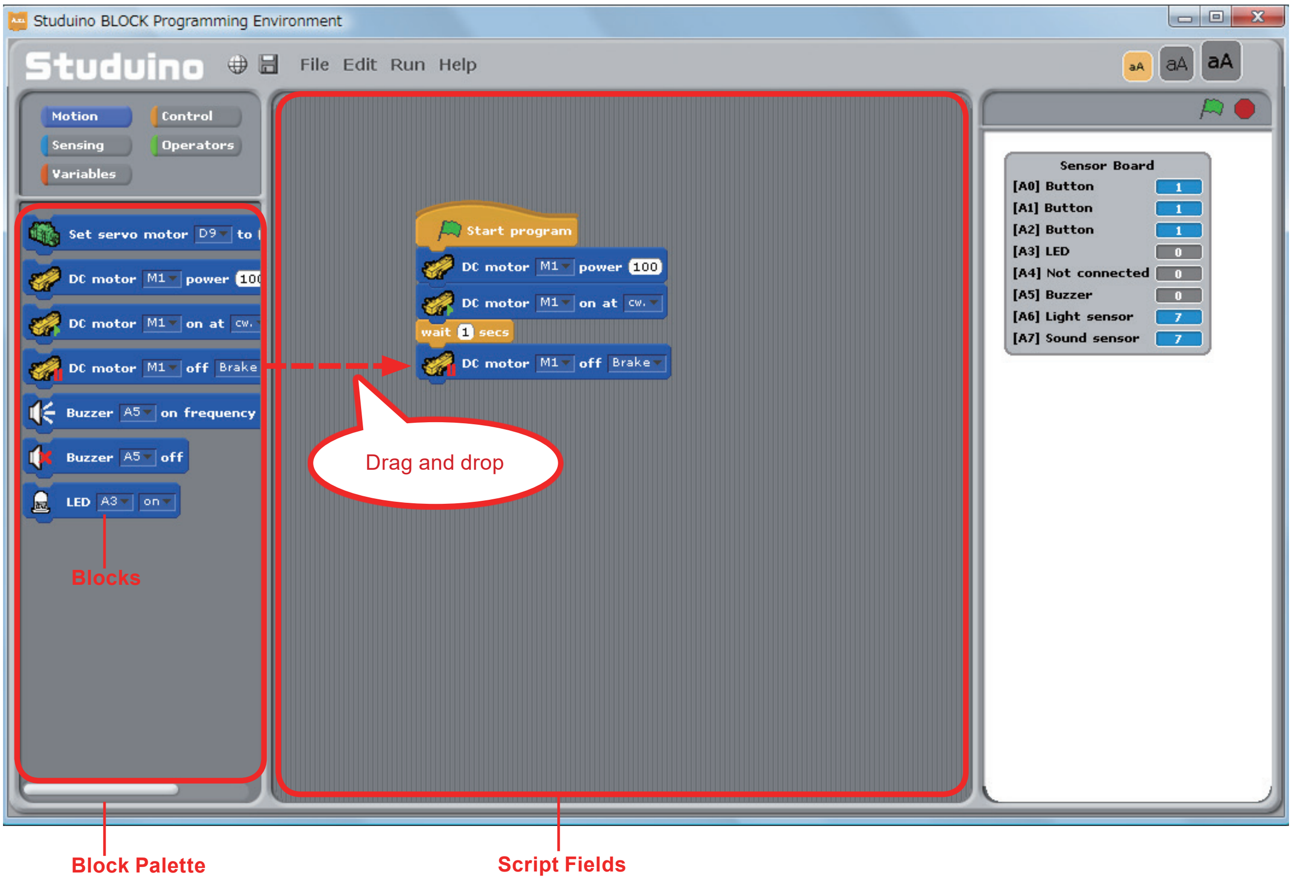
- **Delete a column**

Removes a column and moves other columns to the left.



6. The Studuino Block Programming Environment

6.1. Overview and Features



Start program
DC motor M1 power 100
DC motor M1 on at cw.
wait 1 secs
DC motor M1 off Brake

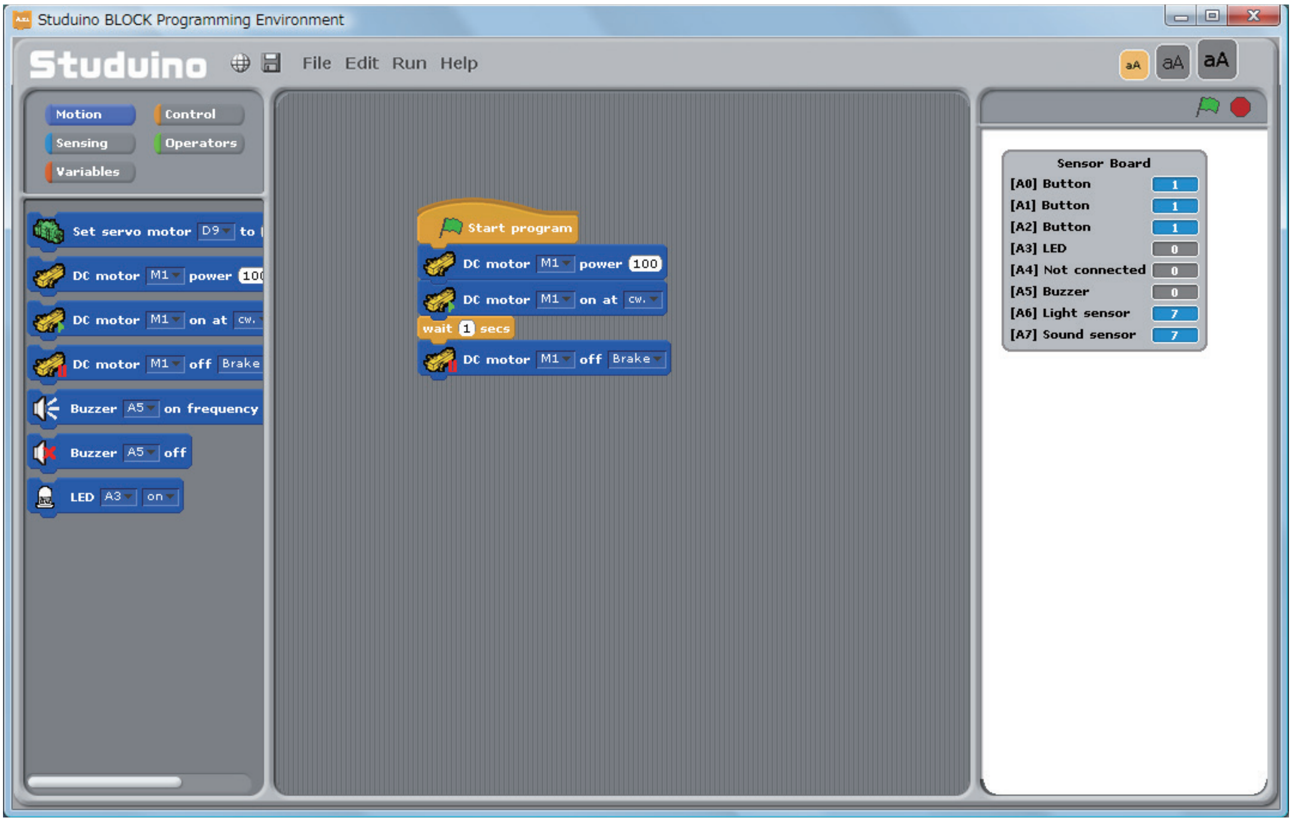
Sensor Board

[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] LED	0
[A4] Not connected	0
[A5] Buzzer	0
[A6] Light sensor	7
[A7] Sound sensor	7

The Studuino Block Programming Environment is a visual programming environment for ArtecRobo, based on the Scratch programming environment developed by the Massachusetts Institute of Technology. Use it to create programs for your Studuino by dragging blocks from the Block Palette and dropping them to connect them to other blocks.

6.2. BPE Robots

6.2.1. Overview and Features



```
Start program
DC motor M1 power 100
DC motor M1 on at cw.
wait 1 secs
DC motor M1 off Brake
```

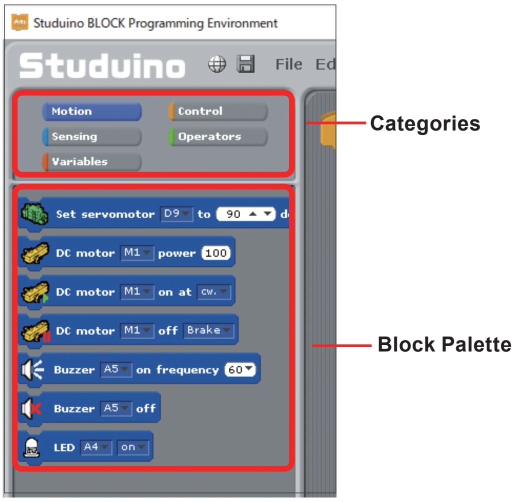
Sensor Board	
[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] LED	0
[A4] Not connected	0
[A5] Buzzer	0
[A6] Light sensor	7
[A7] Sound sensor	7

The Robots build of the Studuino Block Programming Environment was created to make programming your ArtecRobo robot that much easier. Use Test Mode to communicate with your robot in real time as you check its values and see how it works (see **6.2.5. The Main Menu** for details).

Once you've finished programming, you can send your program directly to your ArtecRobo robot by transferring it. You can also convert your program to Arduino language and use the Arduino IDE to edit it (see **Display Arduino Language** in **6.2.5. The Main Menu** for details).

6.2.2. Categories and Block Palettes

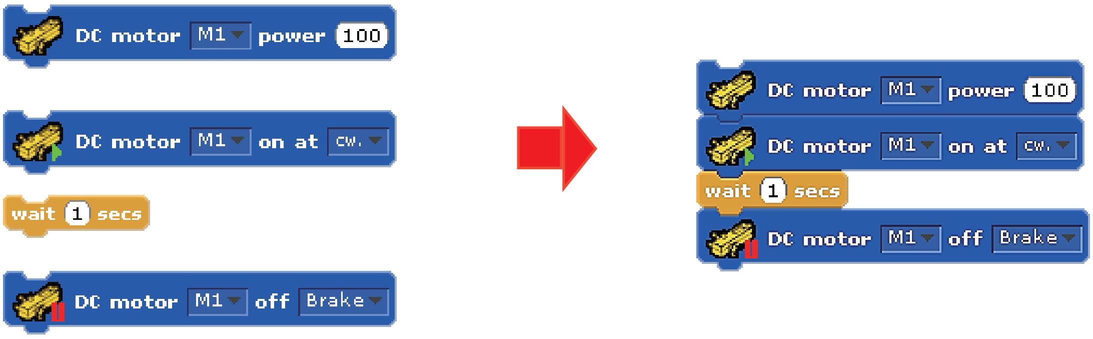
The blocks you'll use to program are divided into Motion, Control, Sensing, Operators, and Variables. You can select a Category of blocks by clicking on its button. Follow along below to learn about the blocks in each Category.



Button	Block
Motion	Control DC Motors, Servomotors, Buzzers, and LEDs
Control	Control basic programming elements such as conditions, functions, and wait processes
Sensing	Look up sensor values
Operators	Control basic and advanced arithmetic and logic operators
Variables	Make and control variables and lists

6.2.2.1. More About Blocks

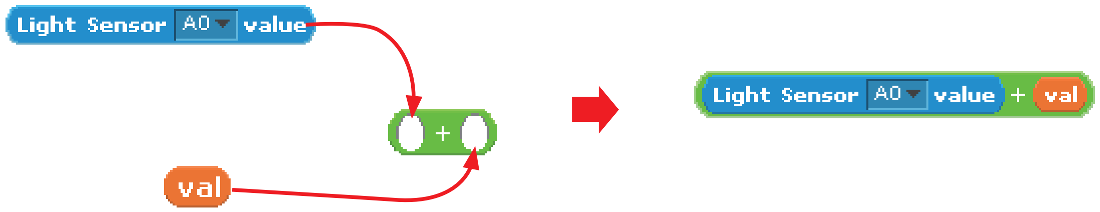
In the Studuino Block Programming Environment there are two kinds of blocks which are divided into those with notches on the top and bottom such as wait 1 secs (process blocks) and those without notches such as Light Sensor A6 value and < (setting blocks). Process blocks are used mainly for actions. These are the ones you connect to create the programs that control your robot.



Blocks connect vertically

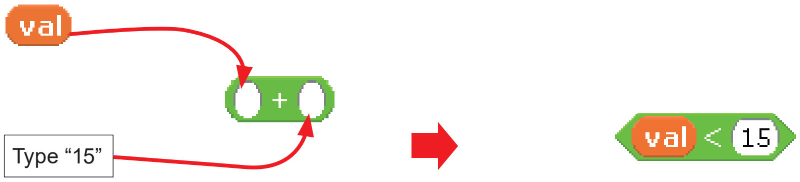
Sets the speed of the DC Motor to 100
And stops it after one second

Blocks with rounded edges such as Light Sensor A6 value return values and are used mainly to modify the settings of other blocks. Hexagonal input boxes like < are used to set conditions and are mainly placed inside of condition blocks like if .



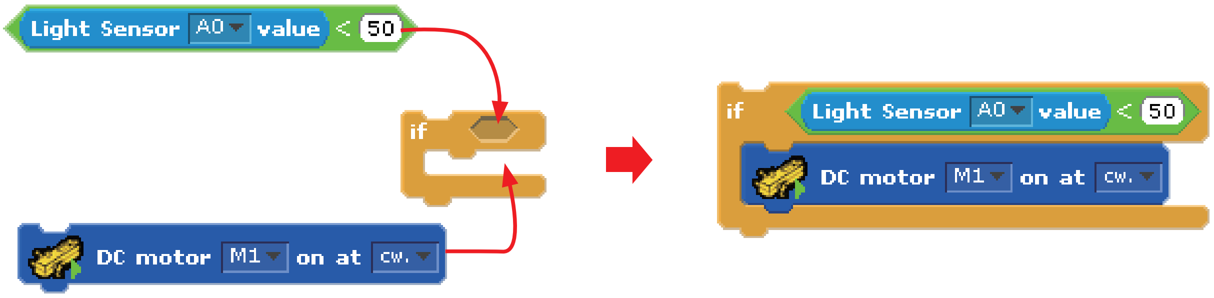
Putting together a Light Sensor value block and a val block…

Creates a block which uses the sum of the Light Sensor and val blocks



Make an operator block that checks the val block against 15 using <…

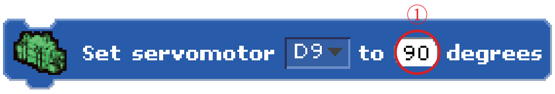
And this operator will check whether val is less than 15

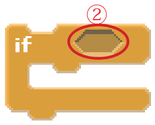


Add a block which checks if the Light Sensor value is less than 50 and a DC Motor block to a branching condition block…

And this condition will make the DC Motor run if the value of the Light Sensor is less than 50

You can make settings for blocks by using the rounded space in ① or the hexagonal space in ② .

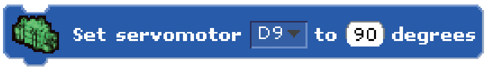




The rounded input box in ① can accept blocks like Light Sensor A6 value or numerical input. The hexagonal space in ② can use blocks like < to change the settings of the block. The following section explains the different types of blocks in the Block Palettes.

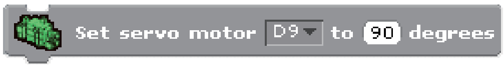
■ Motion Palette Blocks

Blocks in the Motion Palette are used to control the parts of your robot. Blocks for parts without Port Settings configured are grayed out and can't be placed in the Script Field.



Set servomotor D9 to 90 degrees

Connected to Studuino



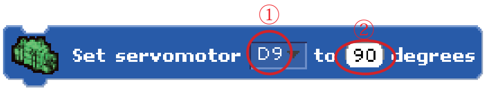
Set servo motor D9 to 90 degrees

Not connected to Studuino

See below for descriptions of each block:

● Servomotor Block

These control the Servomotors connected to your Studuino.

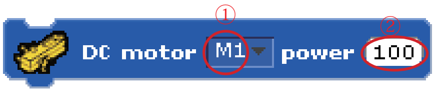


Set servomotor D9 to 90 degrees

Sets the Servomotor on the connector (D2-D12) specified in ① to an angle (0-180) specified in ② . Any setting with a value less than 0 will be set to 0, and any value greater than 180 will be set to 180.

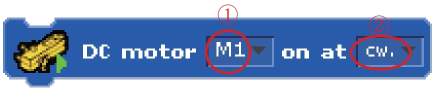
● DC Motor Blocks

These control the DC Motors connected to your Studuino.



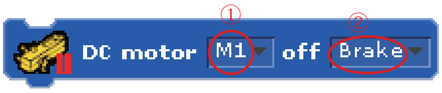
DC motor M1 power 100

Sets the DC Motor on the connector (M1/M2) specified in ① to the speed (0-100) specified in ② . The higher the speed, the faster the DC Motor will rotate. Any setting with a value less than 0 will be set to 0, and any value greater than 100 will be set to 100.



DC motor M1 on at cw.

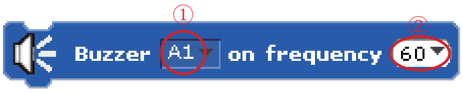
Sets the DC Motor on the connector (M1/M2) specified in ① to the direction (clockwise/counterclockwise) specified in ② .

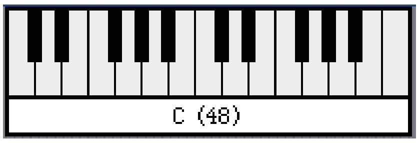


DC motor M1 off Brake

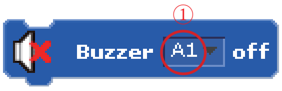
Sets the DC Motor on the connector (M1/M2) specified in ① to the stopping method (Brake/Coast) specified in ② .

- **Buzzer Blocks**



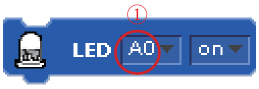


Sets the Buzzer on the connector (A0-A5) specified in ① to the note (frequency) specified in ② . Click ▼ in ② to set the note you would like the Buzzer to play. You can select scales ranging from 48 (C3, 130Hz) to 107 (C8, 4186Hz). You can also select notes ranging from 48 (C3, 130Hz) to 72 (C5, 523Hz) using the on-screen keyboard. If you would like to use scales higher than 72, type them directly into the block by using the keyboard on your PC. Playing notes with the Buzzer may interfere with the operation of a DC Motor connected to M1. If you would like to use a DC Motor connected to M1, use a block to stop the Buzzer before running the DC Motor block.



The combo box in ① is used to stop any Buzzer connected to (A0-A5).

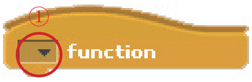
- **LED Blocks**



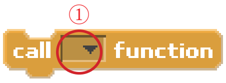
The combo box in ① is used to turn any LED connected to (A0-A5) on or off.

■ **Control Palette Blocks**

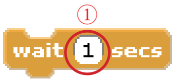
Blocks in the Control Palette allow you to control the flow of your program. See below for descriptions of each block:



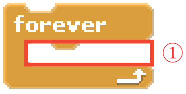
Designates a function. You can name or chose a function in ① . Using non-alphanumeric (A-Z, a-z, 0-9) characters in function, variable, and list names will cause a Build error when transferring your program (see the **Transfer** section in **6.2.5. The Main Menu** for details). Only use alphanumeric characters (A-Z, a-z, 0-9) for function names.



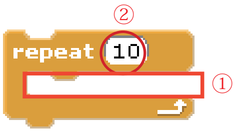
Use ① to call a specific function.



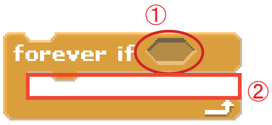
Wait for the number of seconds specified in ① .



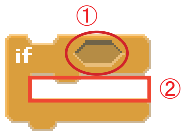
The process inserted into ① will repeat indefinitely.



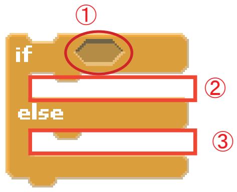
The process inserted into ① will repeat the number of times specified in ② .



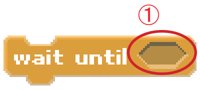
Continuously checks whether the condition in ① has been met and runs the process in ② if it has. The condition in ① is specified using a block from the Operators Palette.



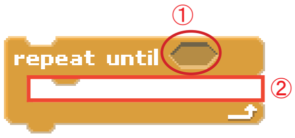
If the condition in ① has been met, the process in ② is run. The condition in ① is specified using a block from the Operators Palette.



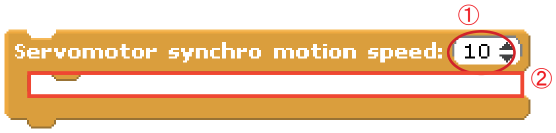
If the condition in ① has been met, the process in ② is run. If the condition hasn't been met, it will run the process in ③ . The condition in ① is specified using a block from the Operators Palette.



Waits until the condition in ① is met. The condition in ① is specified using a block from the Operators Palette.



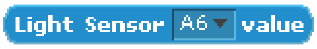
The process in ② repeats until the condition in ① is met. The condition in ① is specified using a block from the Operators Palette.



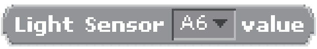
This block sets the Servomotor blocks in ② to the speed set in ① . This block will only work with Servomotor blocks.

■ **Sensing Palette Blocks**

Blocks in the Sensing Palette are used to retrieve sensor values. Blocks for parts without Port Settings configured are grayed out and can't be placed in the Script Field.

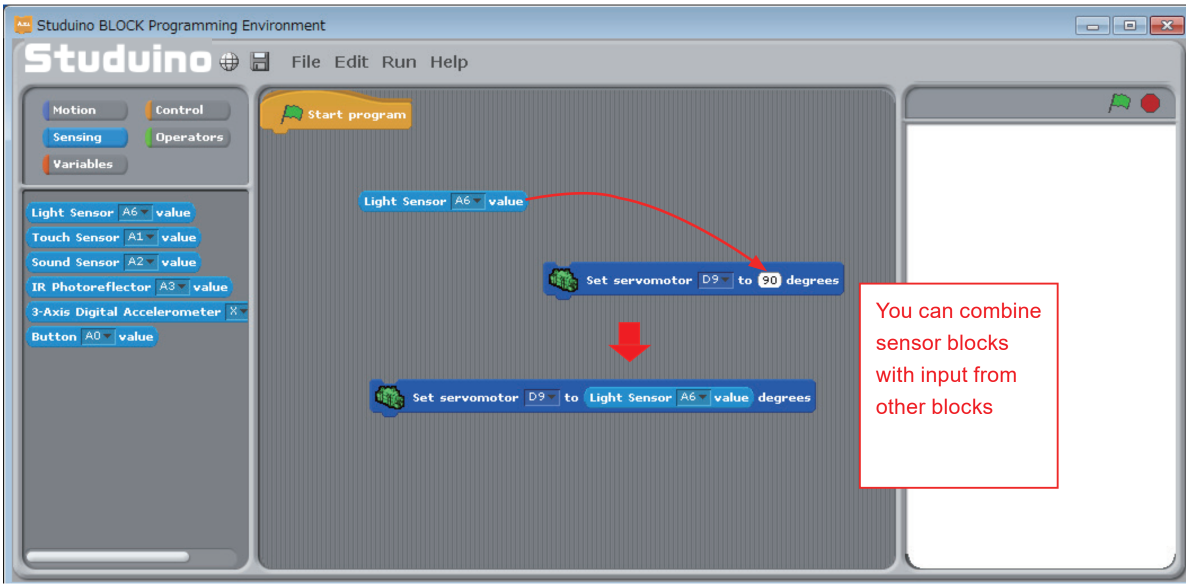


Connected to Studuino



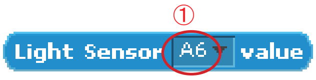
Not connected to Studuino

Sensor blocks can be combined with and use the values of other blocks. The picture below shows a combination of a Servomotor and a Light Sensor block. This combination allows you to change the angles of the motor in response to the amount of light in a room.

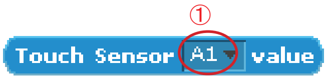


You can combine sensor blocks with input from other blocks

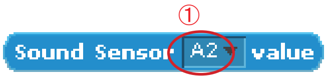
See below for descriptions of each block:



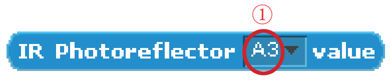
Looks up the value of the Light Sensor using the connector (A0-A7) specified in ①. Light Sensor values have a range of 0 to 100.



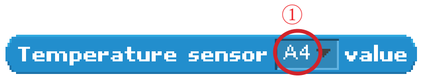
Looks up the value of the Touch Sensor using the connector (A0-A5) specified in ①. Touch Sensors have a value of 0 when pressed and 1 when released.



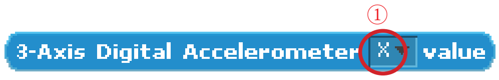
Looks up the value of the Sound Sensor using the connector (A0-A7) specified in ①. Sound Sensor values have a range of 0 to 50.



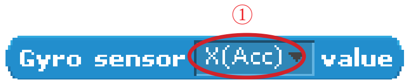
Looks up the value of the IR Photoreflector using the connector (A0-A7) specified in ① . IR Photoreflector values have a range of 0 to 100.



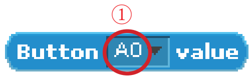
Looks up the value of the Temperature Sensor using the connector (A0-A5) specified in ① . Temperature Sensor values show temperatures in ℃ .



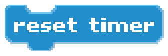
Looks up the X, Y, or Z axis of an Accelerometer. Accelerometer values have a range of 0 to 100.



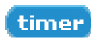
This block looks up your Gyro Sensor's acceleration and angular velocity values. Acceleration values range from -2.0 to 2.0 g (for gravitational acceleration), allowing you to select X(Acc), Y(Acc), or Z(Acc) for the X, Y, or Z axis. Angular velocity values range from -250 to 250 dps (degrees per second), allowing you to select X(Gyro), Y(Gyro), or Z(Gyro) for the X, Y, or Z axis.



Looks up the value of the Push-button (A0-A3) specified in ① . Push-buttons have a value of 0 when pressed and 1 when released.



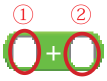
Resets the timer to 0.



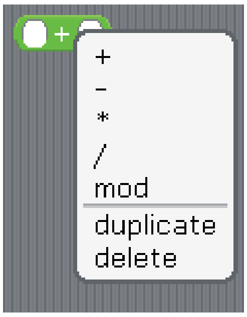
Looks up the value of the timer. Timer values are shown in seconds.

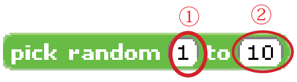
■ Operators Palette Blocks

The Operators Palette contains blocks that perform mathematical operations on the values you input. See below for descriptions of each block:

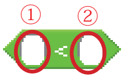


Adds the values set in ① and ② . Arithmetic operator blocks include subtraction (-), multiplication (*), and division (/) blocks. You can select the other arithmetic operators from the context menu by right clicking on the block.

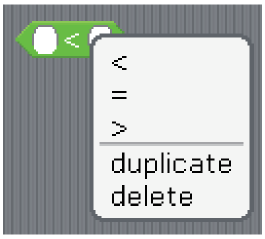




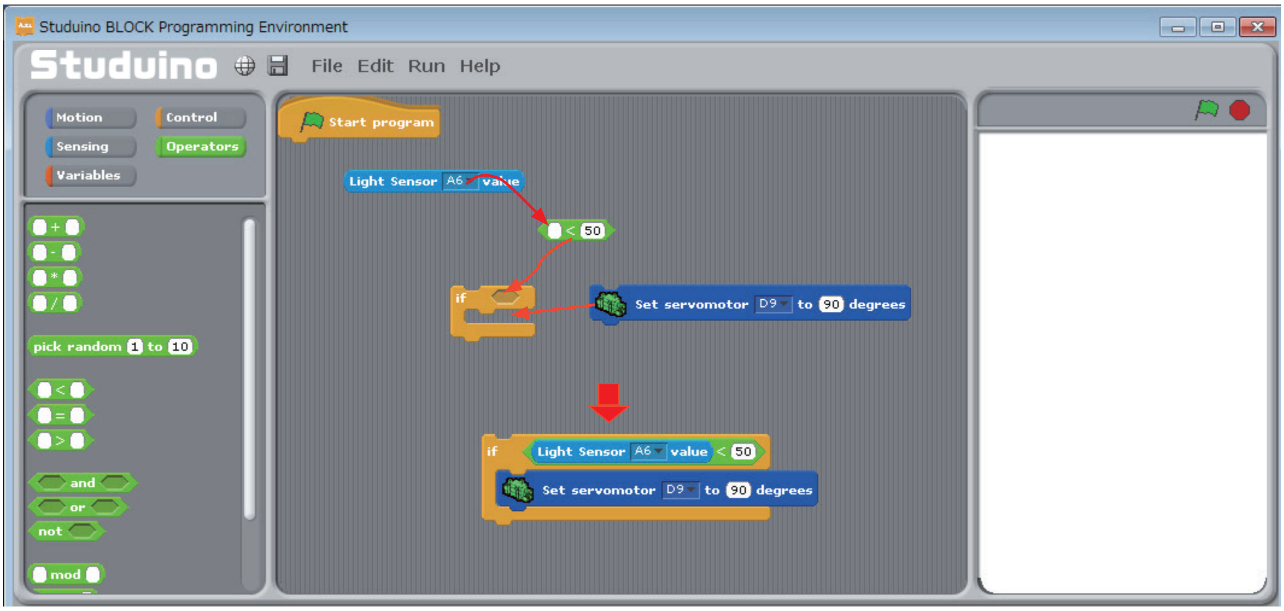
Returns a random number between the values set in ① and ② .

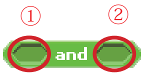


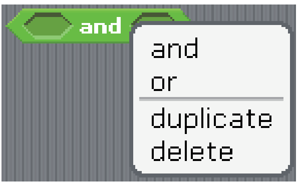
Determines whether the value in ① is less than the value in ② . The other comparative operators find whether the values are equal (=) or whether a value is larger (>). You can select the other comparative operators from the context menu by right clicking on the block.



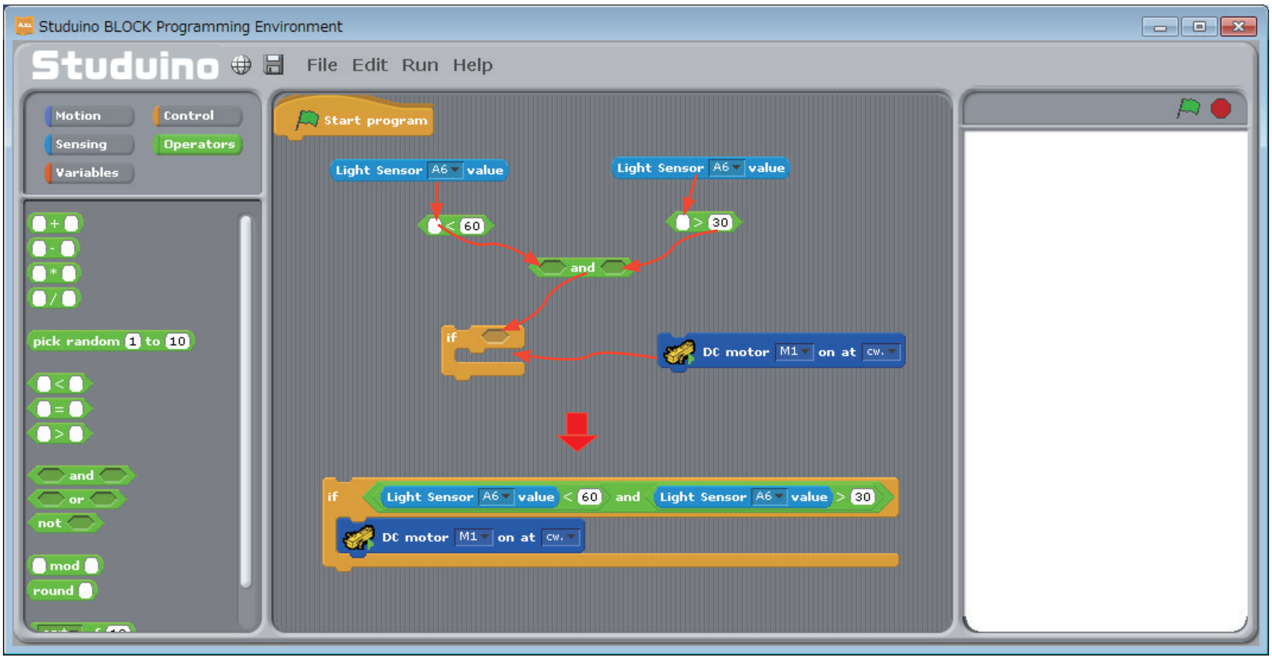
The picture below shows a combination of a Light Sensor, condition, and Servomotor block. You can use this to set the Servomotor angle to 90 degrees if the Light Sensor value dips below 50.

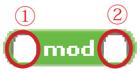




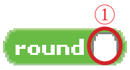


This block is an AND operator using the conditions set in ① and ② . The other logical operator blocks are OR and NOT. You can select the other logical operators from the context menu by right clicking on the block. The picture below shows a combination of a logical operator, condition, and DC Motor block. You can use this combination to make the DC Motor move forward when the Sound Sensor’s values are between 30 and 60.

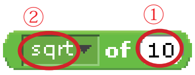


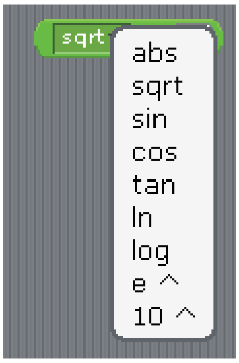


Returns the remainder after dividing value ① by value ② . As with the block, right clicking on this block will allow you to select other arithmetic operators from the context menu.



This block returns the nearest whole number for the value set in ① .



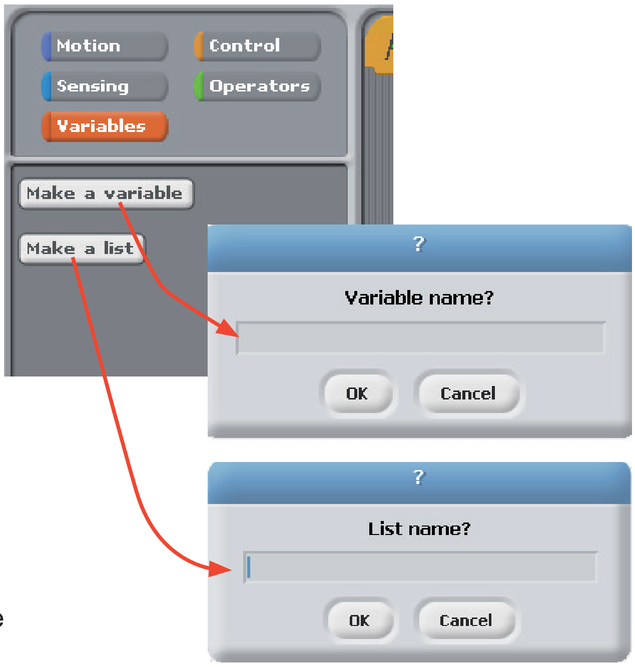


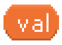
This block returns the value of ① using the arithmetic operation specified in ② . You can choose from absolute values, square roots, trigonometric functions, logarithms, and exponents.

■ Variables Palette Blocks

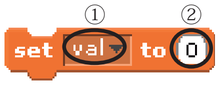
The Variables Palette allows you to create variables and lists. Click the **Make a variable** button and type in a name to make a variable. You can also click the **Make a list** button and type in a name to make a list. Using non-alphanumeric (A-Z, a-z, 0-9) characters in function, variable, and list names will cause a Build error when transferring your program (see the Transfer section in **6.2.5. The Main Menu** for details). Only use alphanumeric characters (A-Z, a-z, 0-9) for function, variable, and list names. Values of variables and lists can be from -3.4028235E+38 to 3.4028235E+38, a maximum of 32 bits (or 4 bytes).

The following section explains the variable blocks in the Variables Palette (for a variable named **val**).

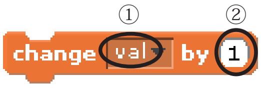




This block looks up the value of the variable.

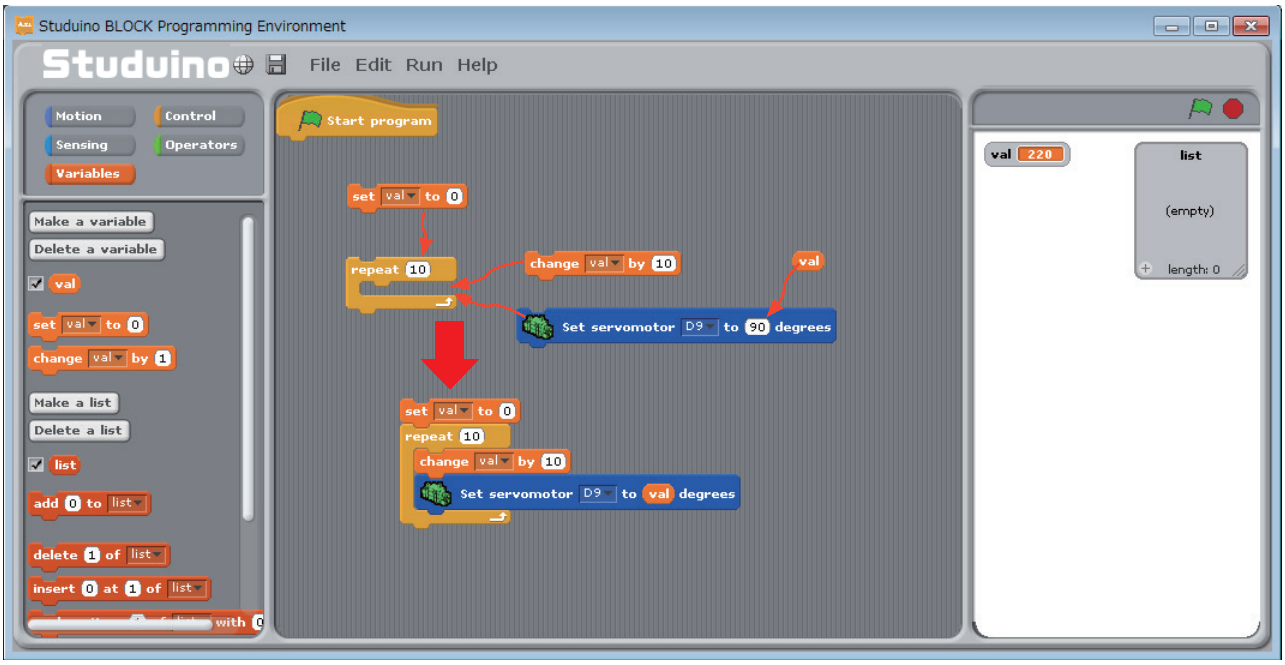


Sets the variable in ① to the value set in ② .



Increases the variable in ① by the value in ② .

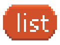
You can make up to 70 unique variables.



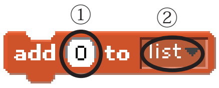
```
Start program

set val to 0
repeat 10
change val by 10
val
Set servomotor D9 to 90 degrees

set val to 0
repeat 10
    change val by 10
    Set servomotor D9 to val degrees
```

The picture above shows a combination of a repeat and a Servomotor block. You can use this to make a program that increases the value in the variable **val** by 10, making the process repeat 10 times and increasing the angle of the motor from 10 to 100 in 10 degree increments.

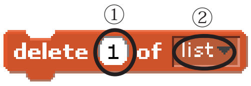
Lists are structured blocks which allow you to add or remove values as you see fit. Lists can hold up to 40 different values. The following section explains the list blocks in the Variables Palette (for a list named **list**).



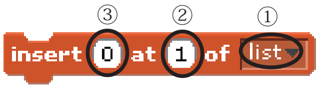
Looks up the value at the top of the list.



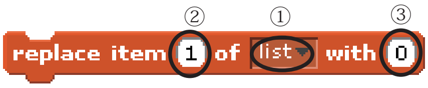
Adds the value in ① to the list in ② .



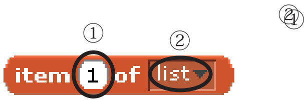
Deletes the value in position ① from the list in ② .



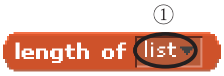
Inserts the value in ③ to list ① at position ② .



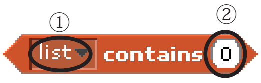
Replaces the value at position ③ in list ① with the value set in ②.



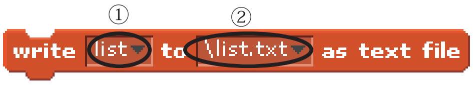
Returns the value at position ② in list ① .



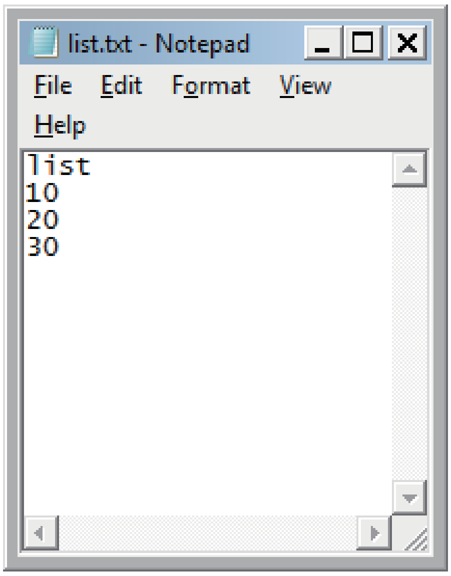
Returns the length of list ① .



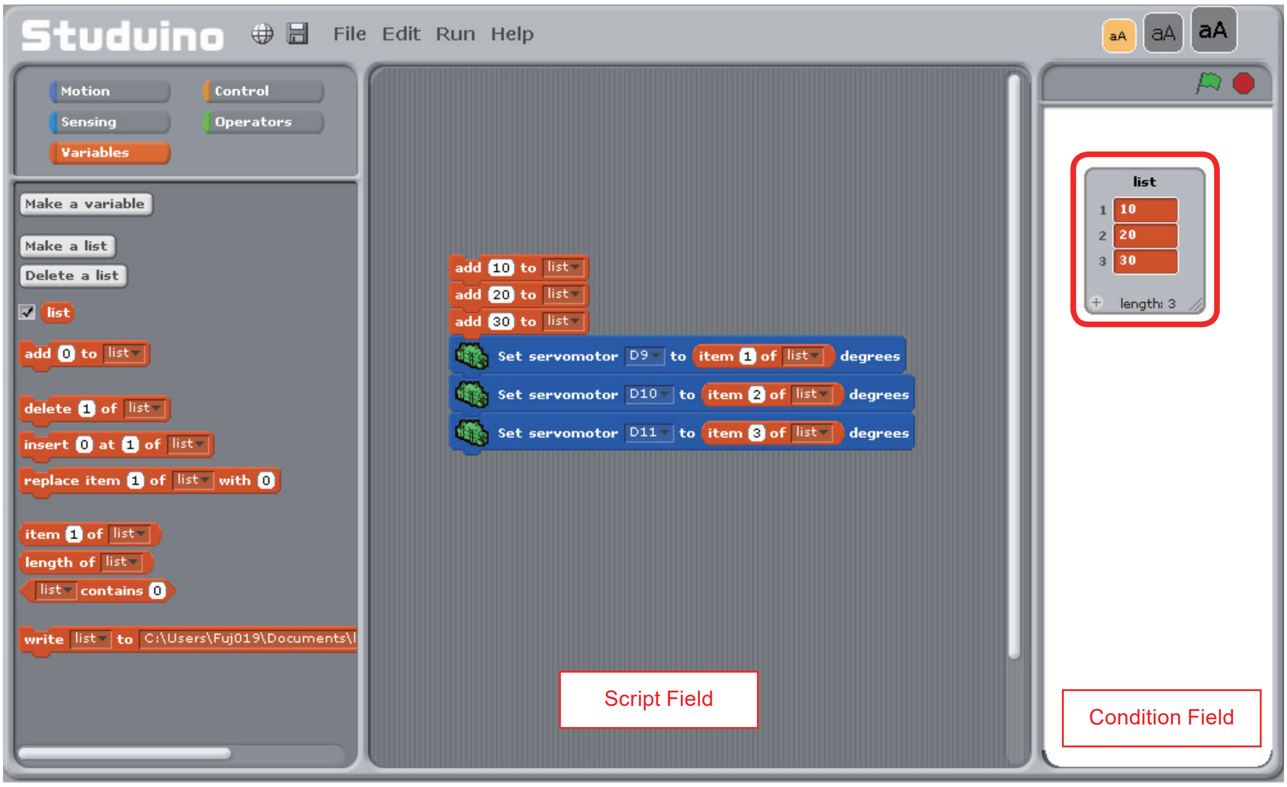
Determines whether list ① includes the value in ② .



Saves the list specified in ① as text to the filename and location specified in ② . As shown below, the content of the list will be shown after the list name.



```
list
10
20
30
```



```
add 10 to list
add 20 to list
add 30 to list
Set servomotor D9 to item 1 of list degrees
Set servomotor D10 to item 2 of list degrees
Set servomotor D11 to item 3 of list degrees
```

The picture above adds the values 10, 20, and 30 to **list**. This process makes a list that includes 10 in the first position, 20 in the second position and 30 in the third position. Using this list, you can set the angles of the Servomotors in the blocks at the bottom. The Servomotor connected to D9 will have its angle set to 10, the Servomotor in D10 to 20, and the Servomotor in D11 to 30 degrees.

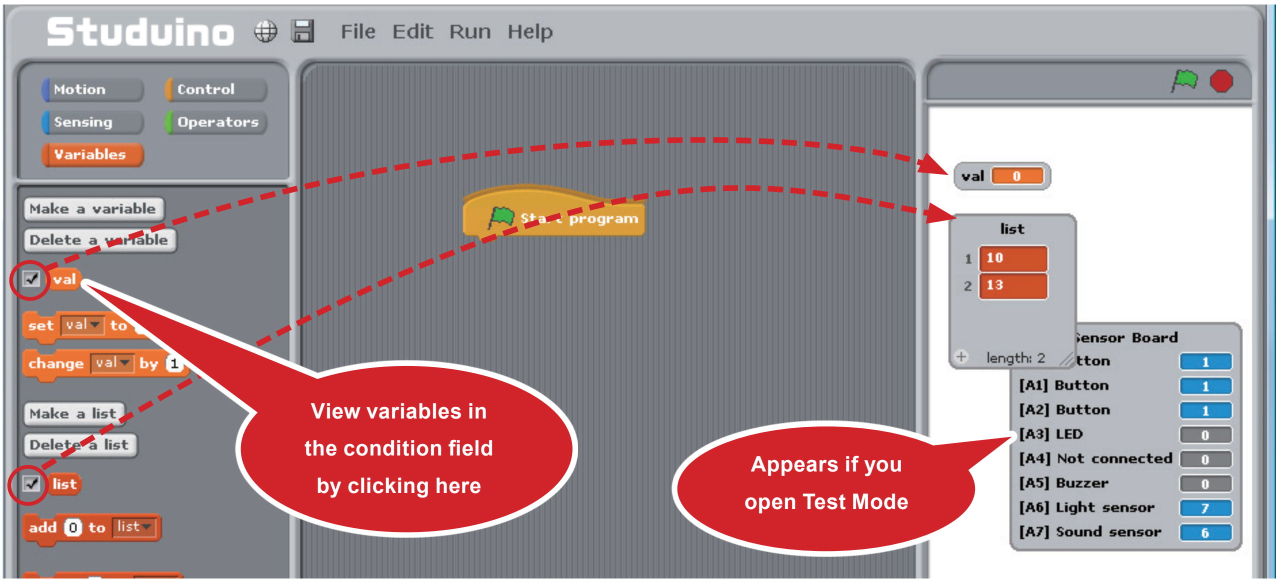
6.2.3. The Script Field

The Script Field is where you drag and connect blocks to create your program. The Studuino Block Programming Environment opens with a Start program Start program block in the Script Field. This block shows the beginning of your program. Any program you make needs to be connected to this block.

6.2.4. The Condition Field

The Condition Field shows sensor values in Test Mode as well as variables and lists. Starting Test Mode will open the Sensor Board, where you can view the values of any sensor connected to your Studuino.

As shown below, you can also view the change in values of any variables or lists in the Condition Field by clicking the checkbox to the left of them.



ensor Board	
tton	1
[A1] Button	1
[A2] Button	1
[A3] LED	0
[A4] Not connected	0
[A5] Buzzer	0
[A6] Light sensor	7
[A7] Sound sensor	6

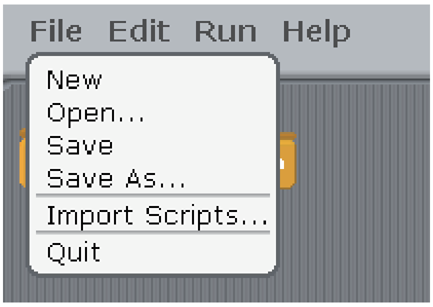
View variables in the condition field by clicking here

Appears if you open Test Mode

6.2.5. The Main Menu

■ **The File Menu**

You can use the File menu to save and load projects.



- **New**

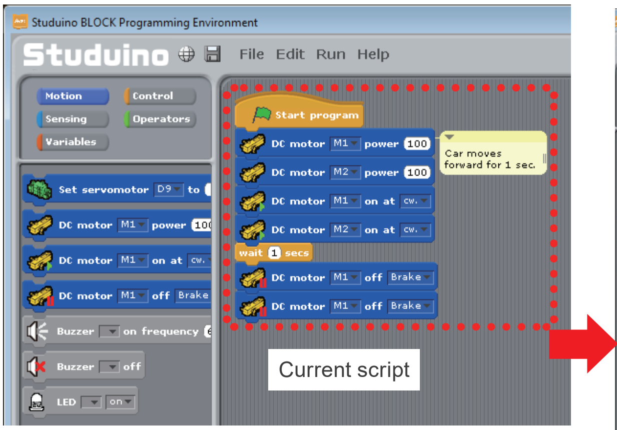
  Start a new project.
- **Open...**

  Open a previously saved project.
- **Save**

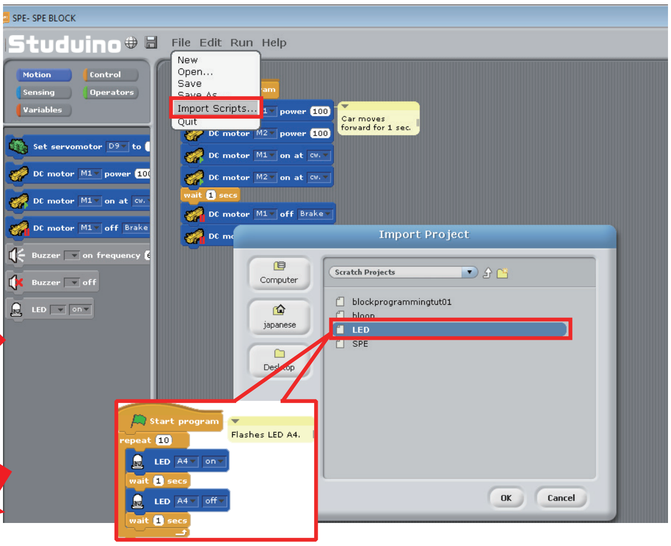
  Save your current project.
- **Save As...**

  Save the project with a specified name.
- **Import Scripts...**

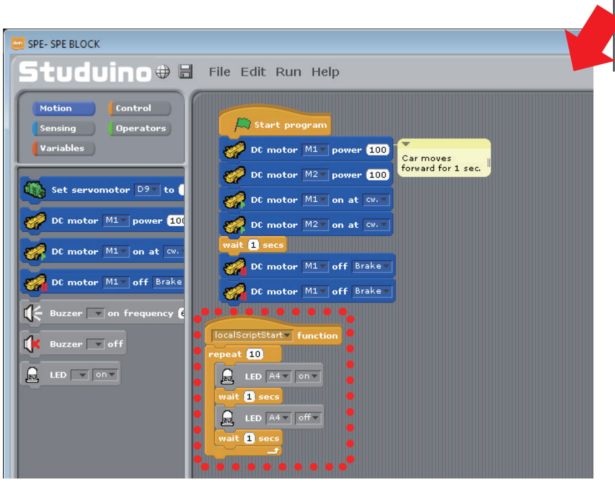
  Load the program from a previous project file.



Current script



Select the Project file you wish to open using the Import Project dialog

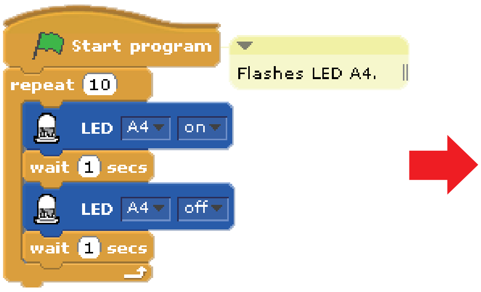


Script added

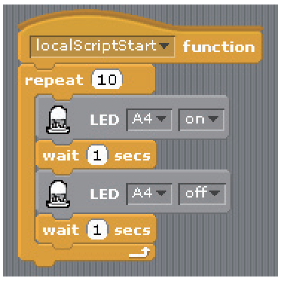
Take note of these three points when loading scripts:

1. This build of the software can’t show more than one yellow Start block at a time, which means the program will be loaded as a function named **localScriptStart**.

2. Any comments in the loaded program will be deleted.

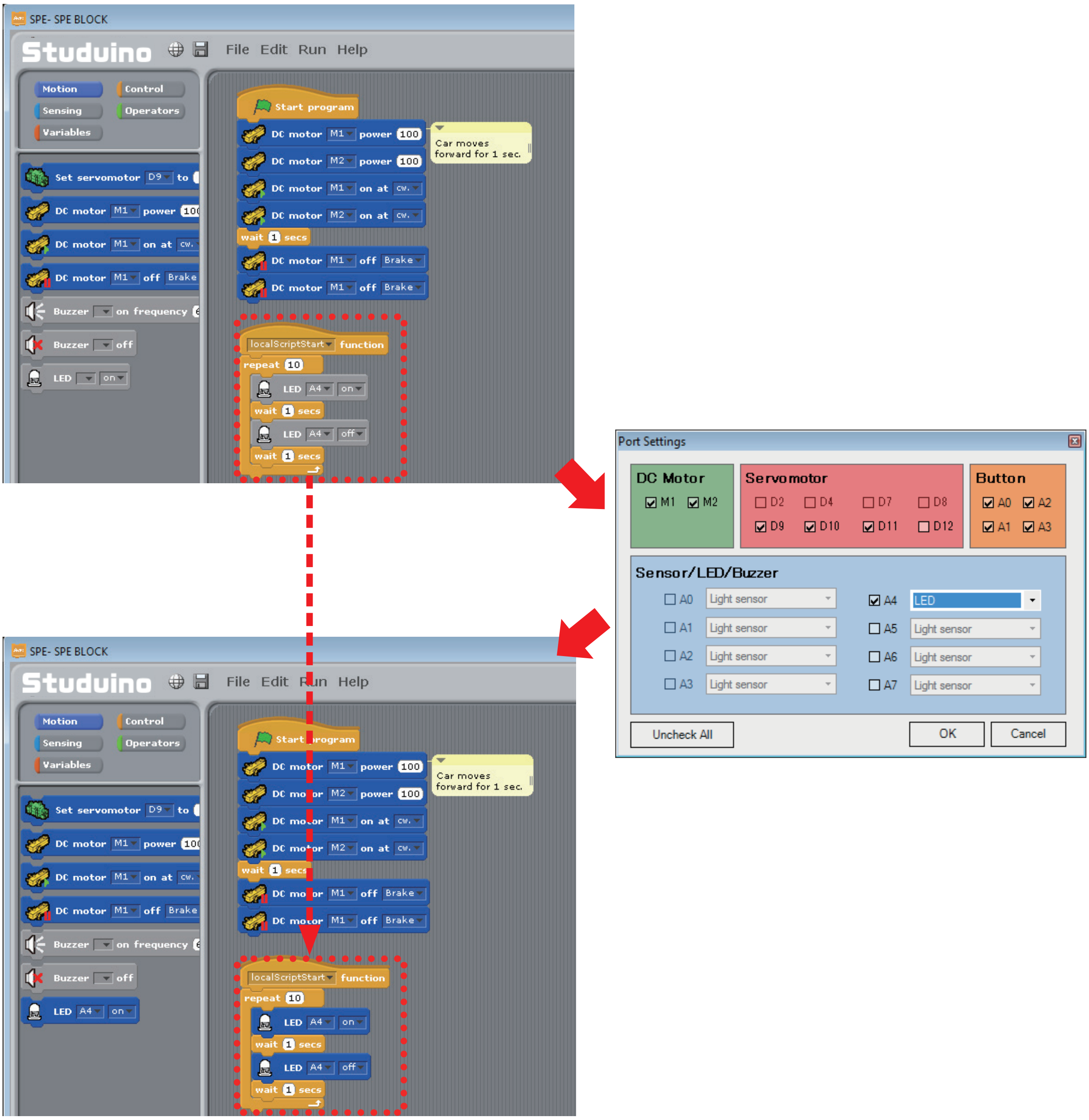


Project file script



Loaded script

3. Port Settings will be changed to those for the loaded script and any blocks for unset parts will be grayed out. These blocks can be activated by setting the corresponding parts in the Port Settings dialog.



```
Start program
DC motor M1 power 100
DC motor M2 power 100
DC motor M1 on at cw.
DC motor M2 on at cw.
wait 1 secs
DC motor M1 off Brake
DC motor M1 off Brake
```

Car moves forward for 1 sec.

```
localScriptStart function
repeat 10
    LED A4 on
    wait 1 secs
    LED A4 off
    wait 1 secs
```

Port Settings

DC Motor: ☑ M1 ☑ M2

Servomotor: □ D2 □ D4 □ D7 □ D8 ☑ D9 ☑ D10 ☑ D11 □ D12

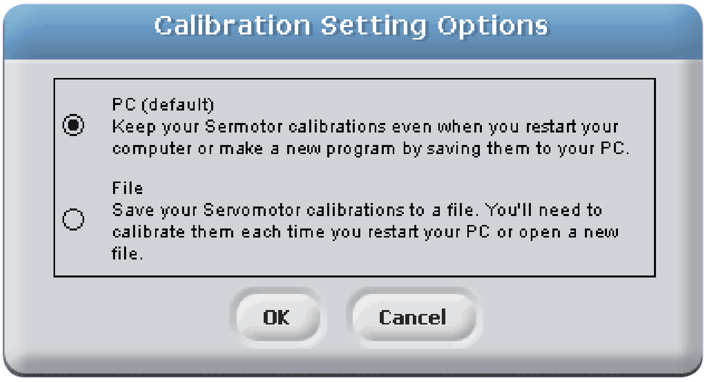
Button: ☑ A0 ☑ A2 ☑ A1 ☑ A3

Sensor/LED/Buzzer

□ A0	Light sensor	☑ A4	LED
□ A1	Light sensor	□ A5	Light sensor
□ A2	Light sensor	□ A6	Light sensor
□ A3	Light sensor	□ A7	Light sensor

- **Calibration Setting Options**

Use this option to specify whether the settings you've made in Motor Calibration are saved to your PC or as a separate file.


Calibration Setting Options

● PC (default)
Keep your Sermotor calibrations even when you restart your computer or make a new program by saving them to your PC.

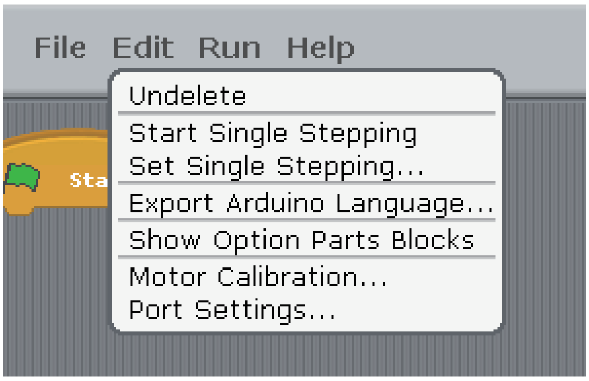
○ File
Save your Servomotor calibrations to a file. You'll need to calibrate them each time you restart your PC or open a new file.

- **Exit**

Closes the Studuino Block Programming Environment.

■ The Edit Menu

The Edit menu allows you to make or edit programs. Only version 2.0 of the Block Programming Environment allows you to use optional parts.

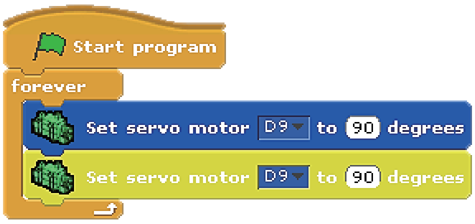


- **Undelete**

  Restores a deleted block.

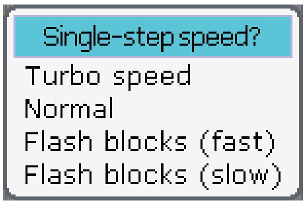
- **Start Single Stepping**

  Starts or stops single stepping for your program. This option starts running the program step by step. When using Single Stepping, the block currently being processed will be highlighted in yellow. You can change how fast steps run using **Set Single Stepping**....


Start program
forever
Set servo motor D9 to 90 degrees
Set servo motor D9 to 90 degrees

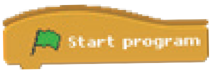
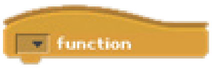
- **Set Single Stepping…**

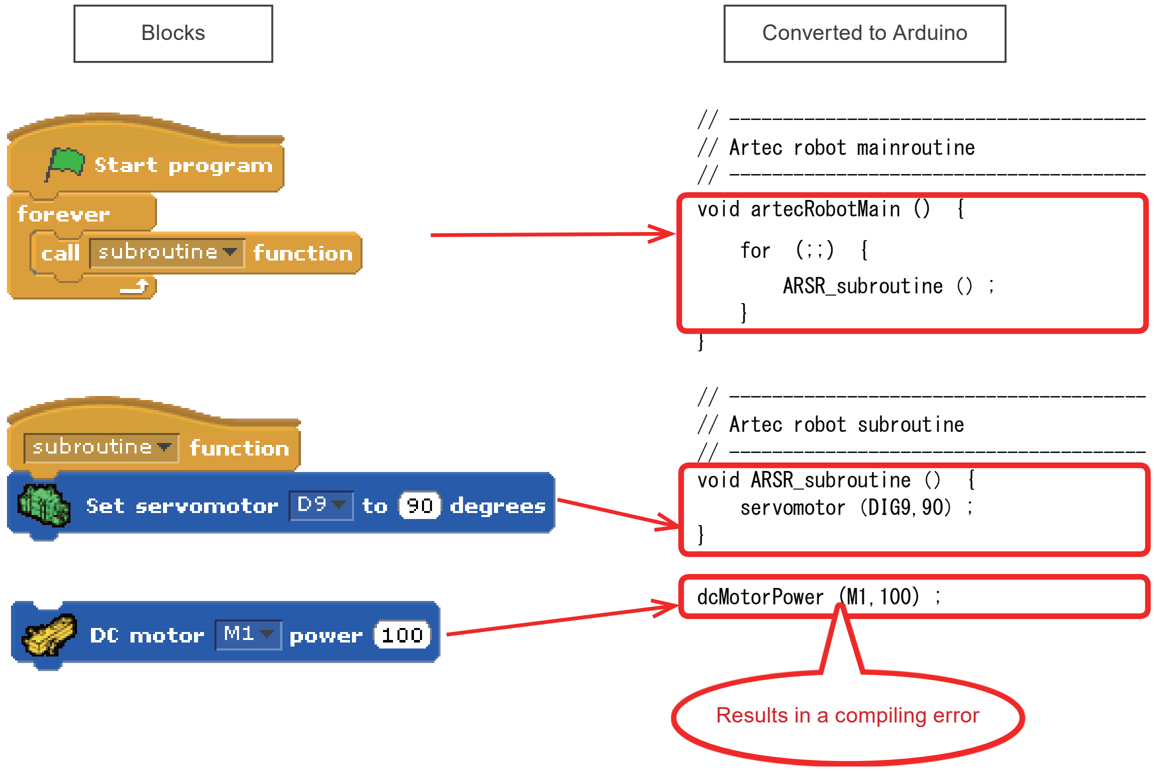
  Use this to change how fast steps are processed. Turbo is the fastest speed, while Flash blocks (slow) is the slowest method of processing steps.



- **Export Arduino Language...**

Converts the program in your Script Field to Arduino language. All blocks in your Script Field will be converted to Arduino language. The source code you export using this feature can be compiled and sent to your Studuino using Arduino IDE.

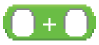
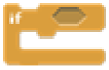
This feature will also convert any blocks unattached to the main  or subordinate  function blocks. Be sure to delete the code for these before compiling using Arduino IDE.



```
// ----------------------------------------
// Artec robot mainroutine
// ----------------------------------------
void artecRobotMain () {
    for (;;) {
        ARSR_subroutine () ;
    }
}
```

```
// ----------------------------------------
// Artec robot subroutine
// ----------------------------------------
void ARSR_subroutine () {
    servomotor (DIG9,90) ;
}
```

```
dcMotorPower (M1,100) ;
```

Blocks which are unattached to functions (orphaned) will still be processed by this feature of the Block Programming Environment, but orphaned code will result in a compiling error in Arduino IDE. Unset values in blocks such as  will be set to **0** when exported to Arduino language, while unset conditions in blocks such as  will be set to **false**.

- **Show Optional Parts (BPE 2.0 only)**

  This option makes blocks for Ultrasonic Sensors, IR Receivers, Temperature Sensors, Gyroscopes, Color Sensors, and Bluetooth Modules available in the Sensing Palette.
- **Hide Optional Parts (BPE 2.0 only)**

  You'll only see this option after selecting **Show Optional Parts**. Selecting this will hide the optional part blocks in the Sensing Palette.

  Follow the link below to download and read **Block Programming Environment Guide (Optional Parts)** for details on these optional parts and their associated blocks.

  http://www.artec-kk.co.jp/studuino/en/studuino_v2.php

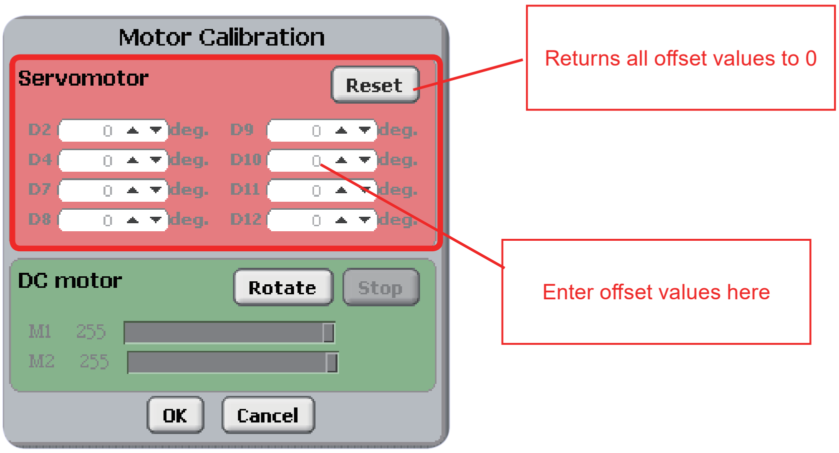
- **Motor Calibration**

Use this feature to adjust the angles of your Servomotors or the speed of your DC Motors and save the settings to a file. Opening this feature will start Test Mode and open the Motor Calibration dialog box.

Studuino Block Programming Environment 1.x uses the following items to calibrate your motors:

▸ **Servomotor Calibration**

When the dialog box opens, all angles for the connected Servomotors are set to 90 degrees by default. Entering an offset value from -15 to 15 degrees will set the angles for the corresponding Servomotor to 90 degrees + the offset value.



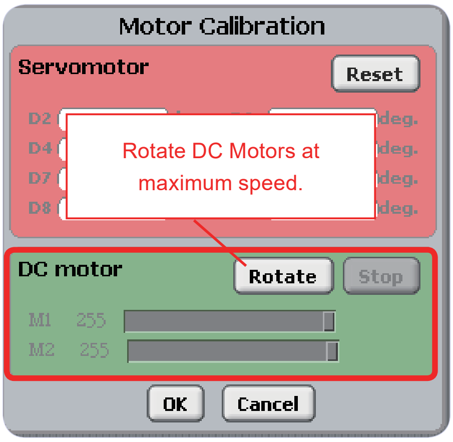
D2 0 deg. D9 0 deg.
D4 0 deg. D10 0 deg.
D7 0 deg. D11 0 deg.
D8 0 deg. D12 0 deg.
M1 255
M2 255

Returns all offset values to 0

Enter offset values here

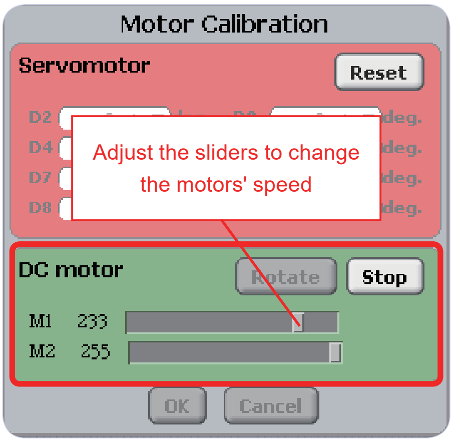
Keep an eye on your Servomotors and adjust the values until you get the correct 90 degree angle. Click the **OK** button once you've finished calibrating your motors. The settings will be saved and the same values will be used every time you start the software.

▸ **DC Motor Calibration**



M1 255
M2 255

Rotate DC Motors at maximum speed.



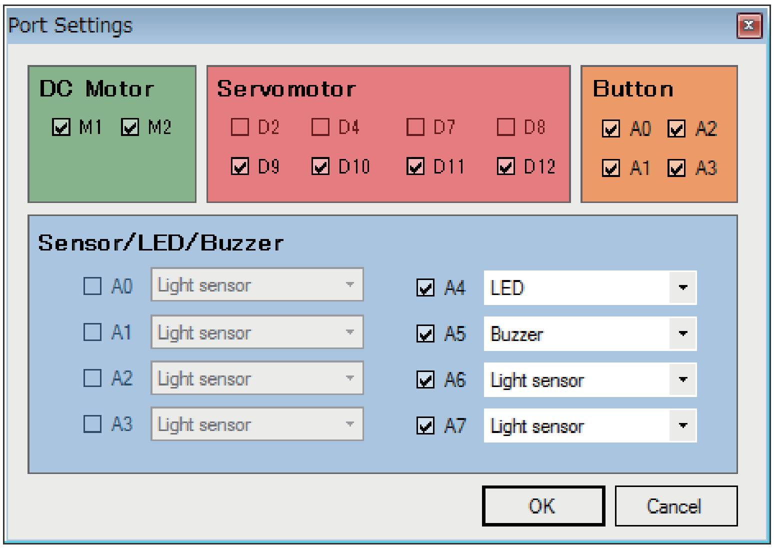
M1 233
M2 255

Adjust the sliders to change the motors' speed

Once the dialog opens, click the **Rotate** button to make your DC Motors rotate at maximum speed. Doing this allows you to adjust the slider until your motors are rotating at identical speeds. Click the **Stop** button to stop your motors. You can now click the **OK** and **Cancel** buttons. Click **OK**. The settings will be saved and the same values will be used every time you start the software.

- **Port Settings...**

Port Settings allows you to set the parts connected to your Studuino in the Studuino Block Programming Environment. Clicking on this will show the Port Settings dialog box.



Port Settings

DC Motor
☑ M1 ☑ M2

Servomotor
☐ D2 ☐ D4 ☐ D7 ☐ D8
☑ D9 ☑ D10 ☑ D11 ☑ D12

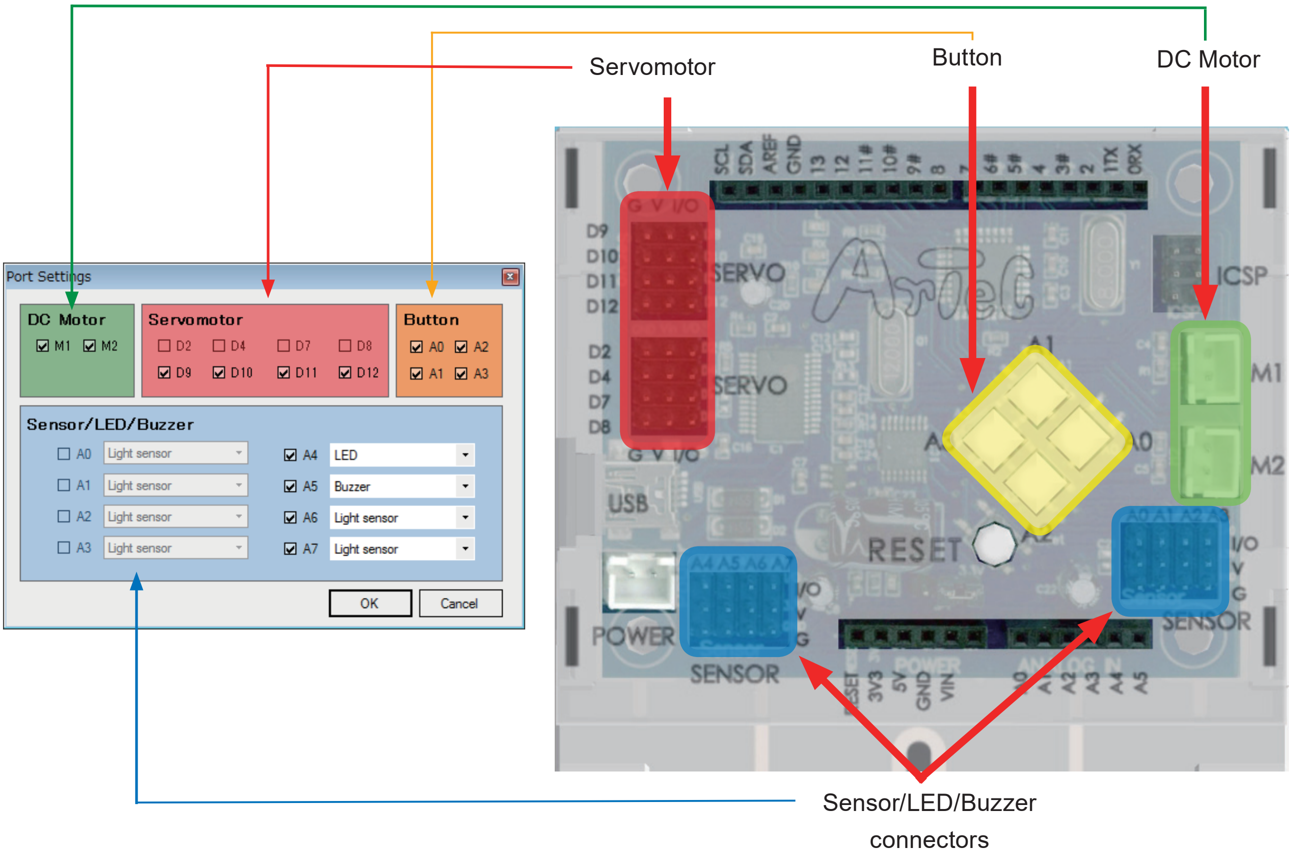
Button
☑ A0 ☑ A2
☑ A1 ☑ A3

Sensor/LED/Buzzer

☐ A0	Light sensor	☑ A4	LED
☐ A1	Light sensor	☑ A5	Buzzer
☐ A2	Light sensor	☑ A6	Light sensor
☐ A3	Light sensor	☑ A7	Light sensor

OK Cancel

As shown in the picture below, this dialog box shows all Studuino connectors and Push-buttons on the Studuino. Be sure to check any which have a part connected to them.



As discussed in 2.3. About Studuino, the following connectors or switches can't be used at the same time:

- DC Motor connector M1 and Servomotor connector D2, D4
- DC Motor connector M2 and Servomotor connector D7, D8
- Push-button A0-A3 and Sensor/LED/Buzzer connector A0-A3

As these combinations also can't be used at the same time in the Port Settings dialog, please uncheck one side while the other is active. For example, when using D2 and D4 for the Servomotor, the checkboxes for D2 and D4 would only become available when DC Motor box M1 is unchecked.

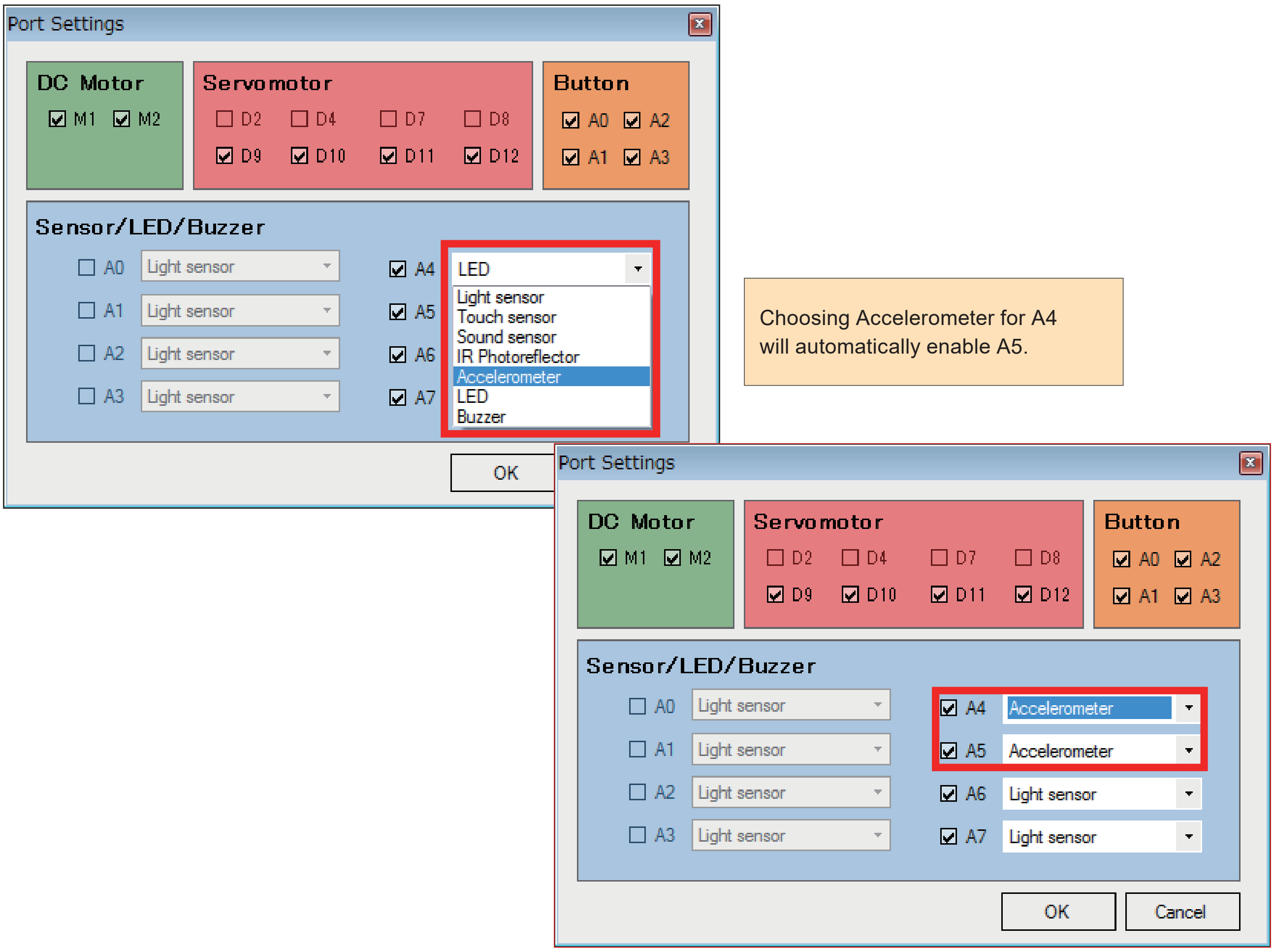
<Caution>

For an Accelerometer, which uses two connectors, all of the corresponding connectors should be checked.

Part	Connector Combination
Accelerometer	A4, A5

Connector combination when multiple connectors are used

When the checkboxes of the corresponding connectors in Connector Combination above are checked, the part connected to the Studuino will be displayed. Checking one box for a part which uses multiple connectors will automatically select the boxes for any other connectors it uses.



Port Settings

DC Motor: ☑ M1 ☑ M2

Servomotor: □ D2 □ D4 □ D7 □ D8 ☑ D9 ☑ D10 ☑ D11 ☑ D12

Button: ☑ A0 ☑ A2 ☑ A1 ☑ A3

Sensor/LED/Buzzer:
□ A0 Light sensor
□ A1 Light sensor
□ A2 Light sensor
□ A3 Light sensor
☑ A4 LED
☑ A5
☑ A6
☑ A7

Choosing Accelerometer for A4 will automatically enable A5.

Port Settings

DC Motor: ☑ M1 ☑ M2

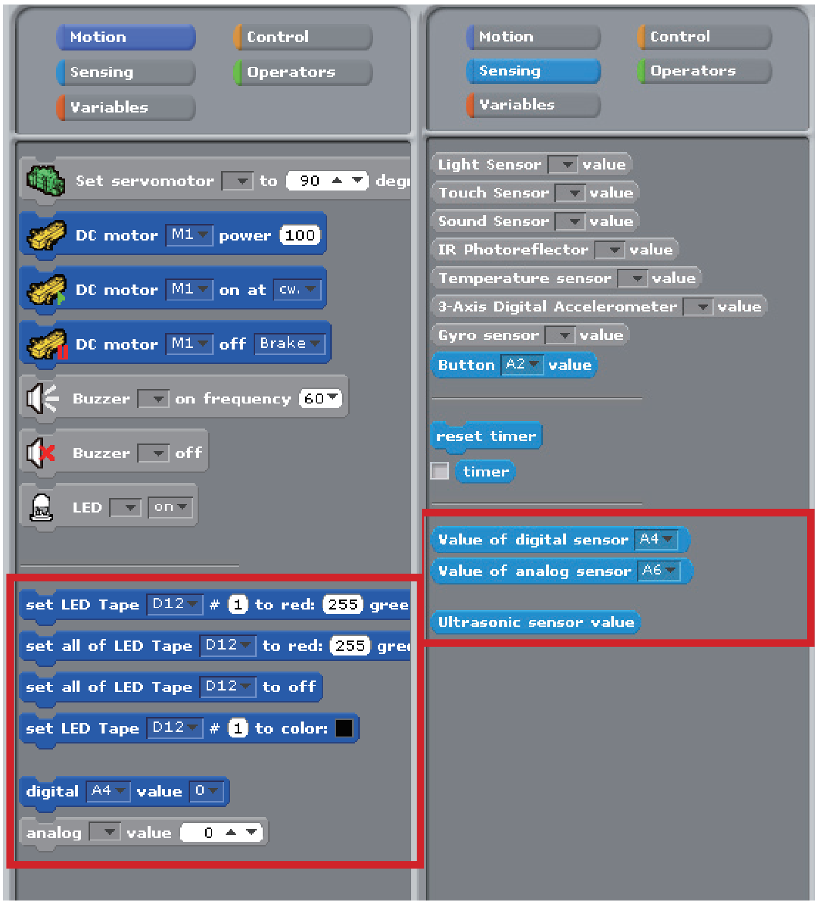
Servomotor: □ D2 □ D4 □ D7 □ D8 ☑ D9 ☑ D10 ☑ D11 ☑ D12

Button: ☑ A0 ☑ A2 ☑ A1 ☑ A3

Sensor/LED/Buzzer:
□ A0 Light sensor
□ A1 Light sensor
□ A2 Light sensor
□ A3 Light sensor
☑ A4 Accelerometer
☑ A5 Accelerometer
☑ A6 Light sensor
☑ A7 Light sensor

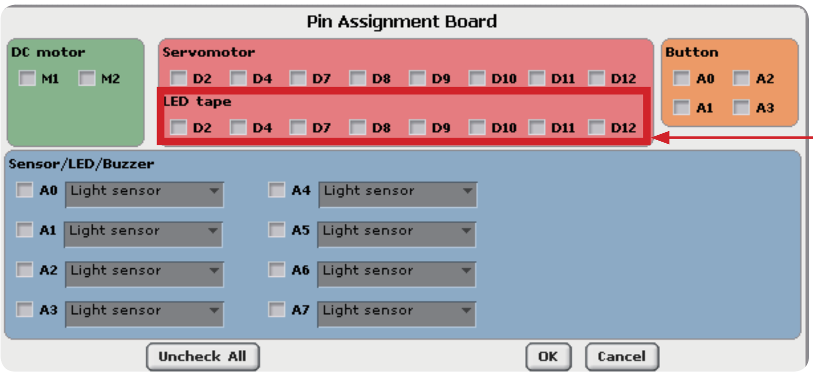
- **Show Extra Blocks**

Choosing this will make extra blocks appear in the the palettes shown below.



- LED Tape Blocks

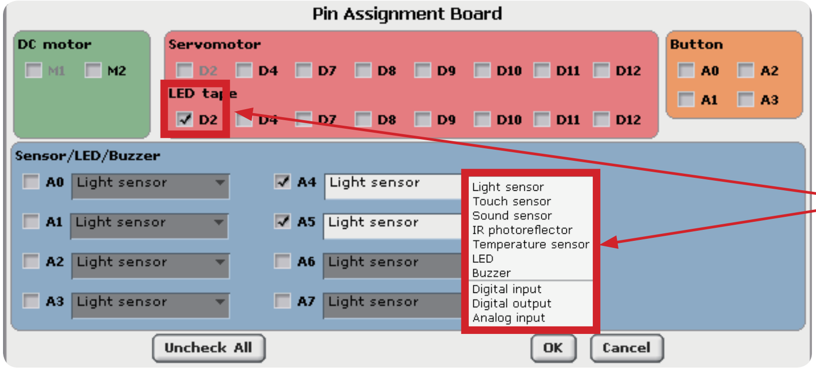
LED Tape consists of several LEDs arranged on a strip of tape, and you can use LED Tape blocks to adjust the color and brightness of each LED. This part can be connected to any Studuino port from D2 to D12, and choosing LED Tape for the corresponding connector in Port Settings will allow you to use the LED Tape blocks.



Choose the connector for your LED Tape

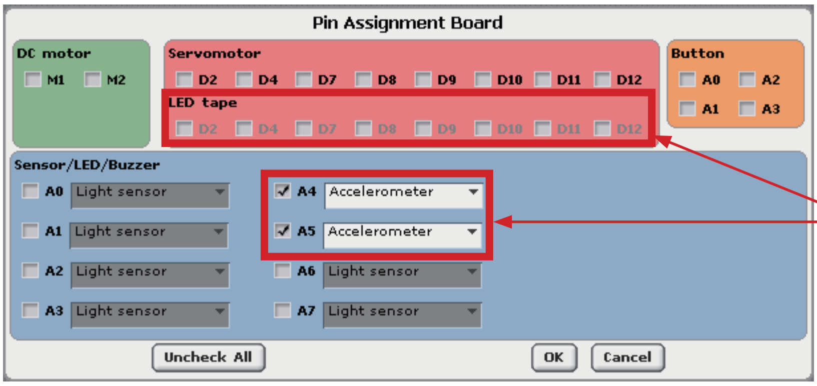
You can't use your LED Tape at the same time as an Accelerometer or Gyro Sensor.

You can't choose an Accelerometer or Gyro Sensor while your LED Tape is selected



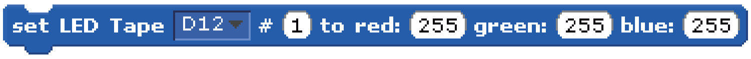
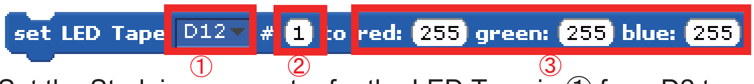
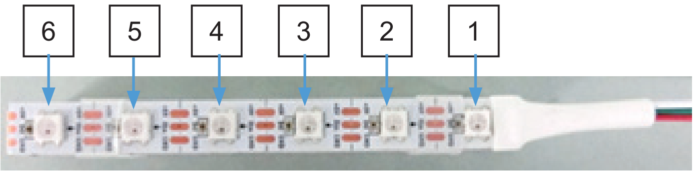
The Accelerometer and Gyro Sensor will be hidden while the LED Tape is selected

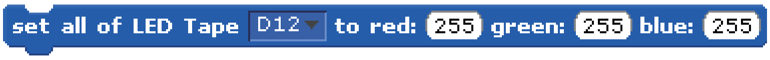
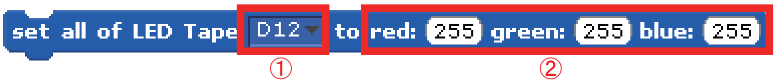
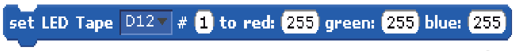
You can't choose your LED Tape while an Accelerometer or Gyro Sensor is selected

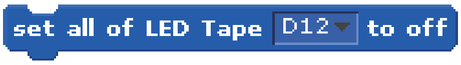
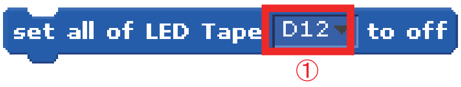


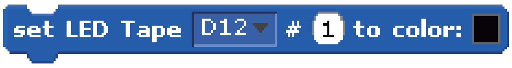
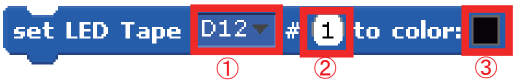
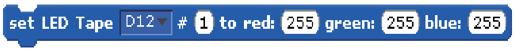
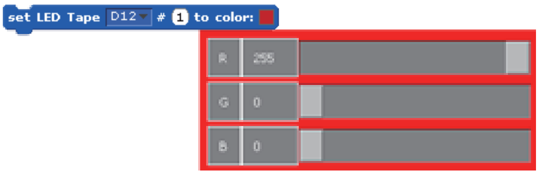
Selecting an Accelerometer or Gyro Sensor will disable the ports for the LED Tape

Read below to find out more about the LED Tape blocks.

Block	 set LED Tape D12 # 1 to red: 255 green: 255 blue: 255
Overview	This block allows you to specify the color of one LED on the LED Tape using RGB values.
Details	 set LED Tape D12 # 1 to red: 255 green: 255 blue: 255 Set the Studuino connector for the LED Tape in ① from D2 to D12 and specify the position of the LED on the tape in ② . Look at the picture below to find the position of each LED.  Use ③ to specify the amount of red, green, and blue for the LED's color in numerical values from 0 to 255.

Block	 set all of LED Tape D12 to red: 255 green: 255 blue: 255
Overview	This block allows you to specify the color of all LEDs on the LED Tape using RGB values.
Details	 set all of LED Tape D12 to red: 255 green: 255 blue: 255 Just like the  set LED Tape D12 # 1 to red: 255 green: 255 blue: 255 block, use ① to select the connector for the LED Tape and ② to specify the color of every LED on the LED Tape.

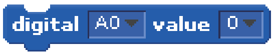
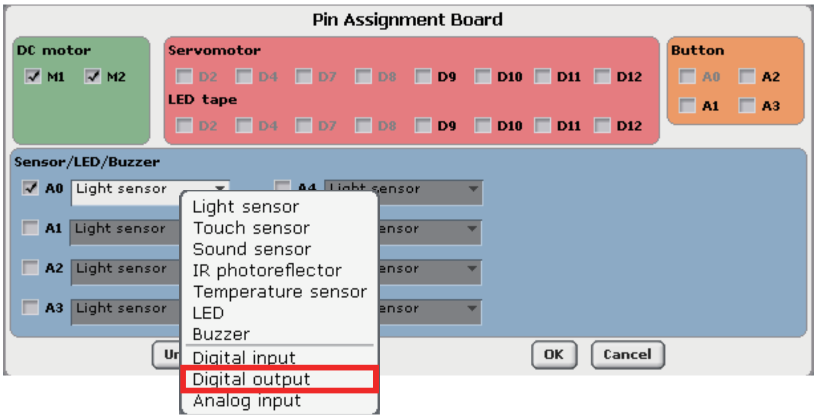
Block	 set all of LED Tape D12 to off
Overview	Turn off all LEDs on the LED Tape.
Details	 set all of LED Tape D12 to off Use ① to specify the connector of the LED you wish to turn off from D2 to D12.

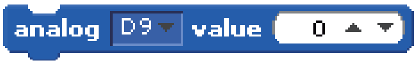
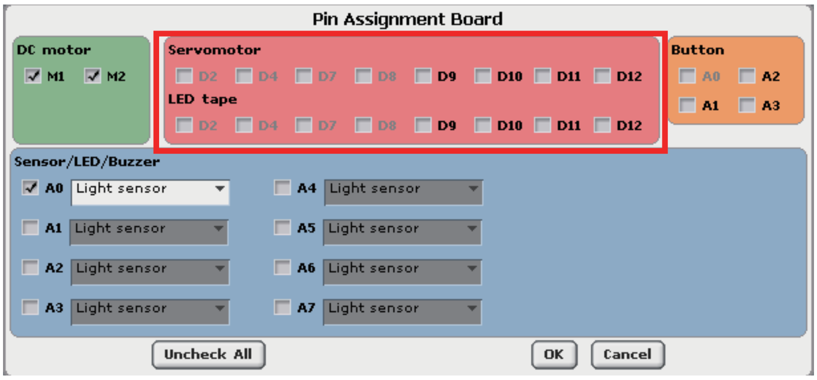
Block	 set LED Tape D12 # 1 to color:
Overview	Blink the specified LED on the LED Tape.
Details	 set LED Tape D12 # 1 to color: Just like the  set LED Tape D12 # 1 to red: 255 green: 255 blue: 255 block, set the Studuino connector for the LED Tape in ① from D2 to D12 and specify the position of the LED on the tape in ② . Use the sliders in ③ to specify the RGB values of the LED's color as shown below.  set LED Tape D12 # 1 to color:

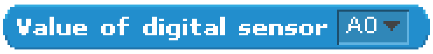
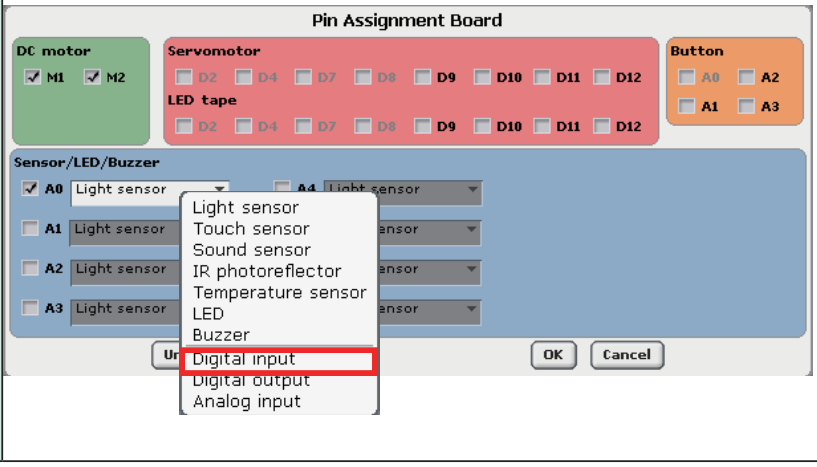
Note that there will be a nearly one second delay when turning the LED Tape's LEDs on or off in Test Mode. While the blocks and your Studuino do exchange data through the USB cable while in Test Mode, it takes some time to send the large amount of data required for the tape's settings.

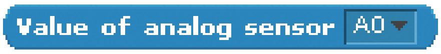
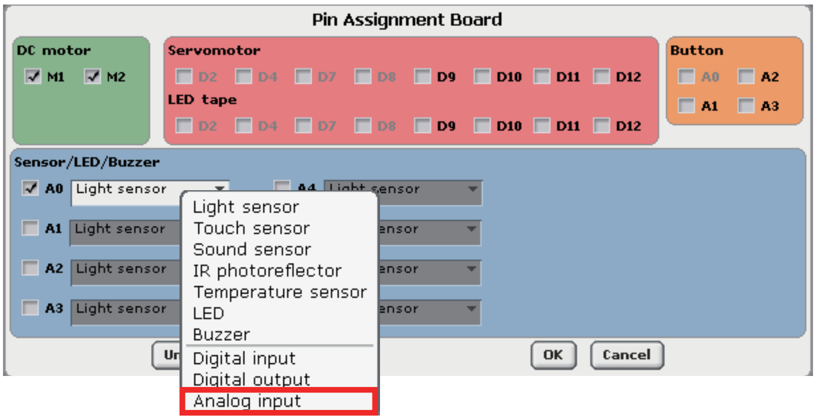
This delay disappears after transferring your program to the Studuino, due to there no longer being a need to transmit the settings for your LED Tape.

General Output Blocks

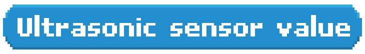
Block	
Overview	Outputs a digital signal.
Details	This block uses any Studuino connector from A0 to A5 as a digital output pin to set a value of 0 or 1 for the connected part. When using a general digital output block, open Port Settings and choose **Digital output** for port A0-A5 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 LED tape □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button □ A0 □ A2 □ A1 □ A3 Sensor/LED/Buzzer ☑ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor Light sensor Touch sensor Sound sensor IR photoreflector Temperature sensor LED Buzzer Digital input Digital output Analog input OK Cancel

Block	
Overview	Outputs an analog signal.
Details	This block uses any Studuino connector from D9 to D11 as an analog output pin to set a value of 0 to 255 for the connected part. When using a general analog output block, open Port Settings and uncheck Servomotor boxes D2 to D12 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 LED tape □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button □ A0 □ A2 □ A1 □ A3 Sensor/LED/Buzzer ☑ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor □ A5 Light sensor □ A6 Light sensor □ A7 Light sensor Uncheck All OK Cancel

Block	
Overview	Looks up a digital signal.
Details	This block uses any Studuino connector from A0 to A5 as a digital input pin to find the value of the connected part.These values are either 0 or 1. When using a general digital input block, open Port Settings and choose **Digital input** for port A0-A5 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 LED tape □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button □ A0 □ A2 □ A1 □ A3 Sensor/LED/Buzzer ☑ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor Light sensor Touch sensor Sound sensor IR photoreflector Temperature sensor LED Buzzer Digital input Digital output Analog input OK Cancel

Block	
Overview	Looks up an analog signal.
Details	This block uses any Studuino connector from A0 to A7 as an analog input pin to find the value of the connected part.These values are from 0 to 1023. When using a general **analog input** block, open Port Settings and choose **Analog input** for port A0-A7 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 LED tape □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button □ A0 □ A2 □ A1 □ A3 Sensor/LED/Buzzer ☑ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor Light sensor Touch sensor Sound sensor IR photoreflector Temperature sensor LED Buzzer Digital input Digital output Analog input OK Cancel

▸ **Ultrasonic Sensor Block**

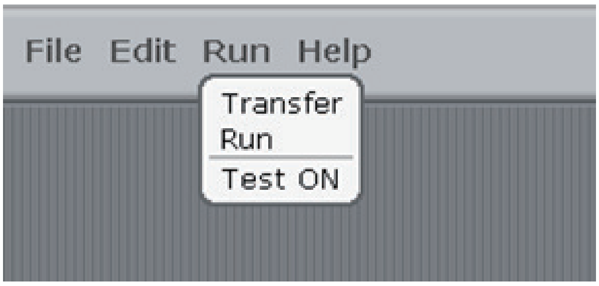
Block	 Ultrasonic sensor value
Overview	Looks up the value of the Ultrasonic Sensor.
Details	Read **7. Ultrasonic Sensors** of the **Studuino Block Programming Environment Guide (Optional Parts)** to learn how to use this block. http://www.artec-kk.co.jp/studuino/docs/en/Studuino_tutorial_block_3.pdf

- **Hide Extra Blocks**

This item is only available in the menu once you've chosen Show Extra Blocks. It hides the extra blocks.

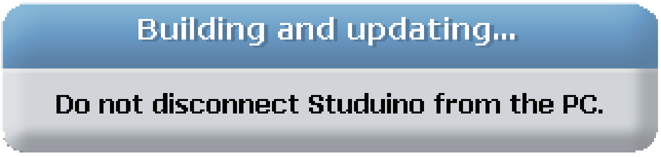
■ The Run Menu

The Run menu allows you to link with your Studuino while making programs.



- **Transfer**

Transfers a program you’ve made to your Studuino. You’ll see the message below when transferring.



Building and updating...

Do not disconnect Studuino from the PC.

Do not disconnect your Studuino from the PC when transferring a program.

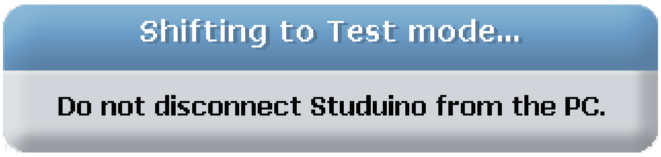
- **Run**

  Run executes any program you've transferred. You’ll only see this once you’ve successfully transferred a program.

- **Test ON/OFF**

  You can use Test Mode to communicate with your Studuino and control parts in real time. It will appear as **Test ON** when Test Mode is not on and **Test OFF** if Test Mode is running. Make sure your PC and Studuino are connected before starting Test Mode.

  This dialog appears if you open Test Mode.



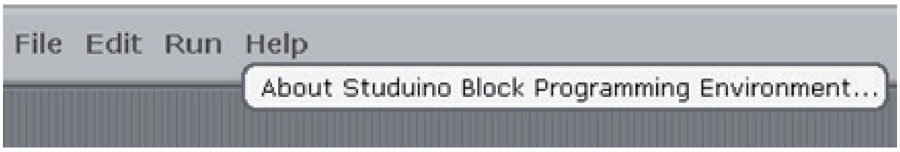
Shifting to Test mode...

Do not disconnect Studuino from the PC.

Do not disconnect your Studuino from the PC when transferring a program.

■ **Help Menu**

The Help menu contains information about the Studuino Block Programming Environment.

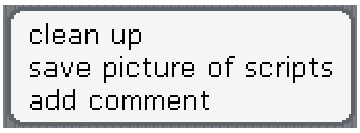


- **About Studuino Block Programming Environment…**

Transfers a program you've made to your Studuino. You'll see the message below when transferring.

6.2.6. The Context Menu

Right clicking on blocks in the Script Field or on the Script Field itself will bring up a context menu.



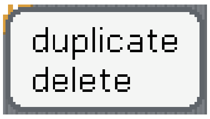
- **Clean Up**

  Arranges blocks in the Script Field.
- **Capture Image**

  Saves an image of the blocks in the Script Field in .gif format.
- **Add Comment**

  Adds a comment to your program. Dragging the resulting comment to any block will append the comment to that block.

Right-clicking on any block aside from the block will bring up a context menu.



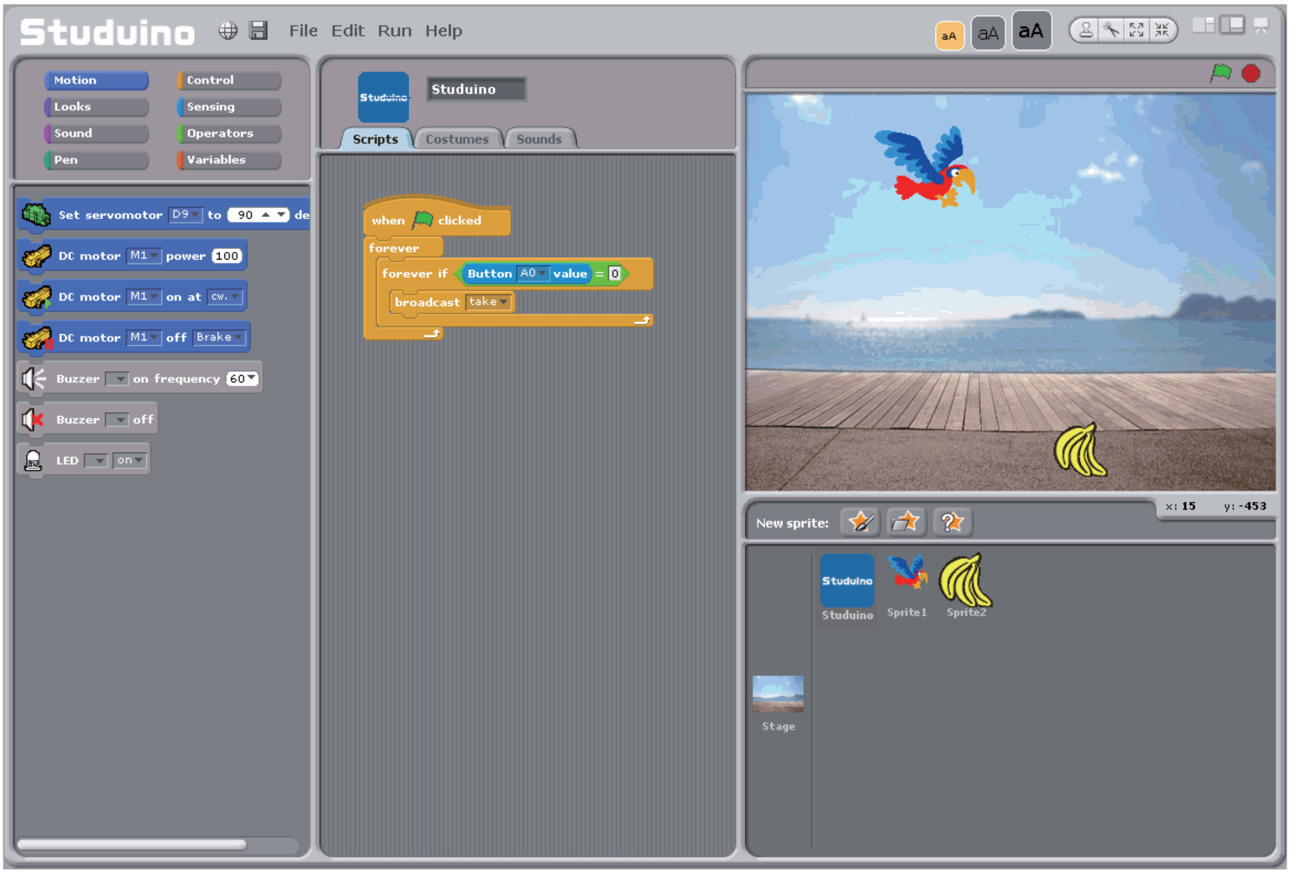
- **Duplicate**

  Duplicates a block or set of blocks.
- **Delete**

  Deletes a block or set of blocks.

6.3. BPE Characters

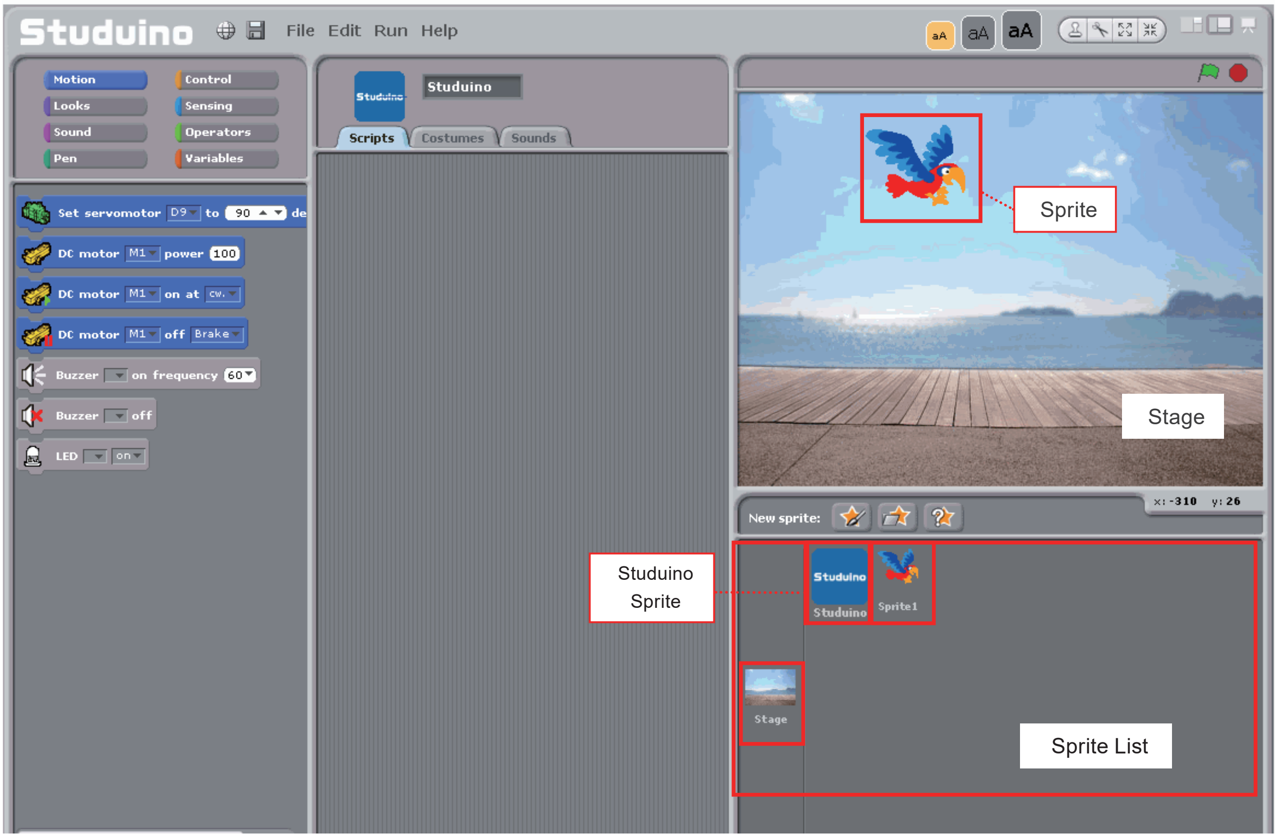
6.3.1. Overview and Features



```
when clicked
forever
  forever if Button A0 value = 0
    broadcast take
```

The Characters build of the Studuino Block Programming Environment allows you to create games and animations that can link with your Studuino. The games you make can be used to control your robot or controlled by the values of your sensors. You can also interact with PC networks called Meshes. A Mesh allows you to connect to a copy of the software running on another computer and create games with network connectivity. You can also load the programs you've made in the Robots build of the Studuino Block Programming Environment or Scratch 1.4, meaning you can link your Studuino with games you've made in Scratch.

6.3.2. Programming



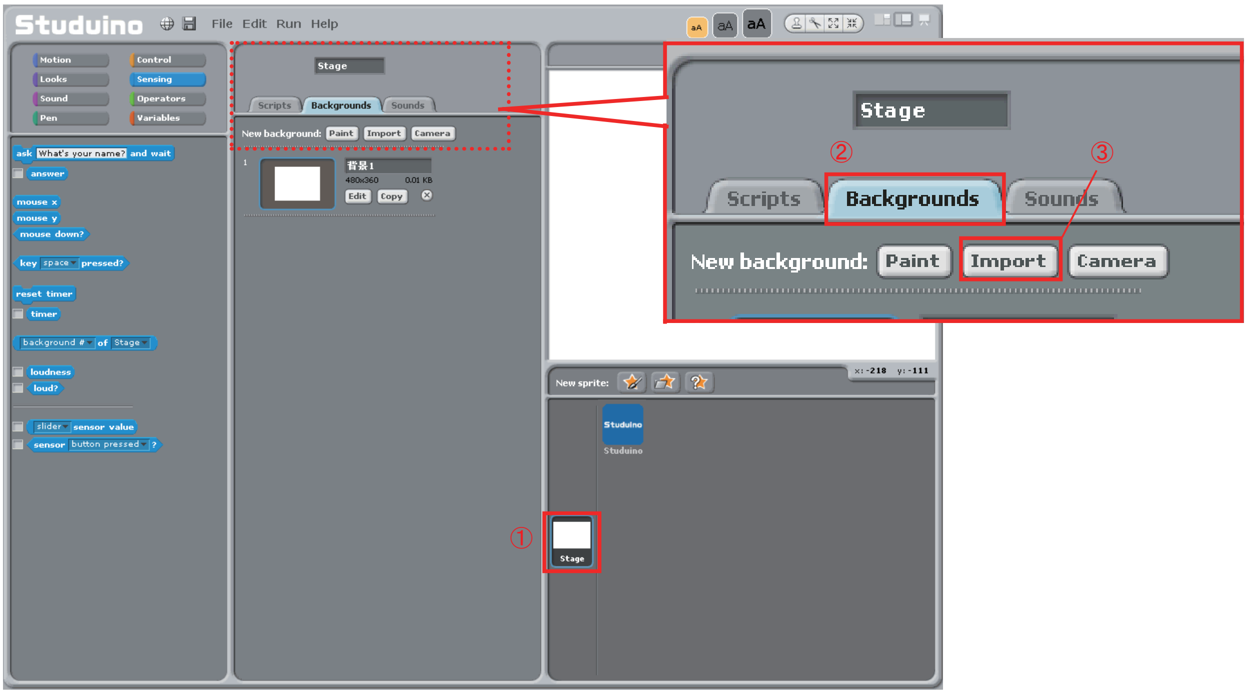
The Sprites in this software are your game characters, and the space they move on is called the Stage. You can program Sprites and the Stage to do different things. You can find the icons for Sprites and Stages in the Sprite List to the lower right of the screen.

The Studuino Sprite in the Sprite List is only used for controlling your Studuino and won't be shown on the Stage.

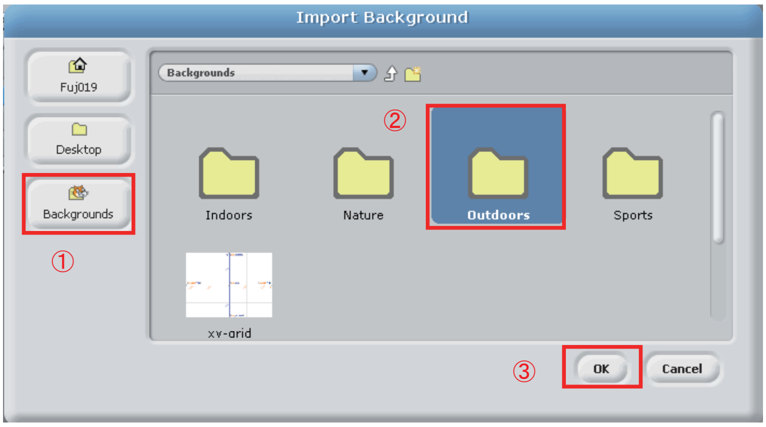
Follow the steps on the next page to get an overview and learn how to program with this software.

6.3.2.1. Adding a Stage

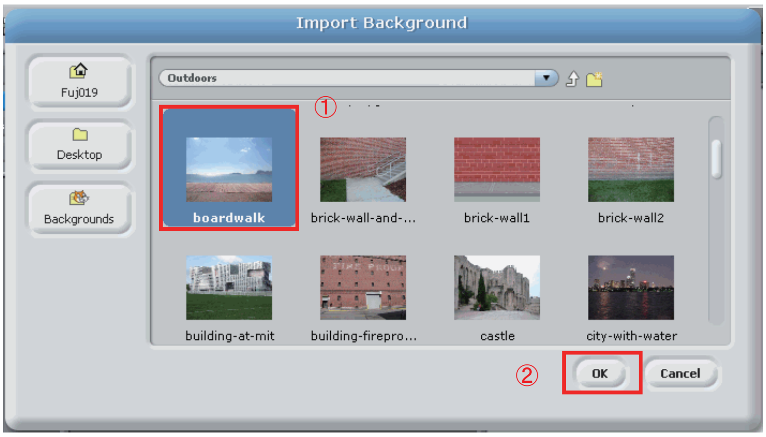
Click the **Stage** icon (①) in the Sprite List. Next, click the **Backgrounds** tab (②) and click the **Import** button (③).



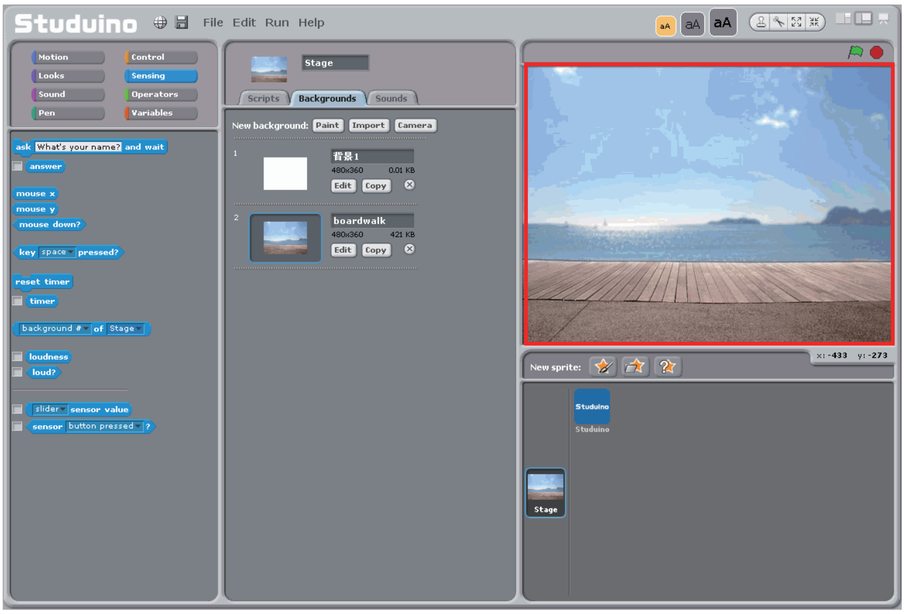
This will bring up the **Import Background** window, giving you a selection of folders to choose from. Click the **Backgrounds** button (①), choose the **Outdoors** folder (②), and click the **OK** button (③).



Choose **boardwalk** (①) from the list of pictures and click the **OK** button (②).

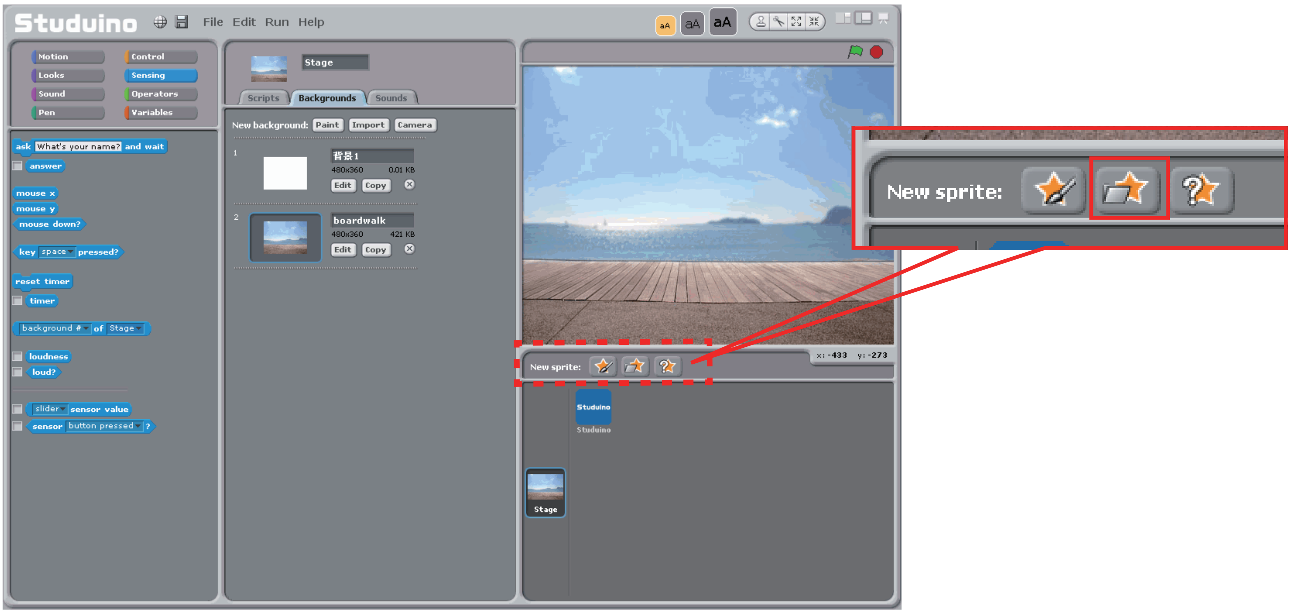


Now you've added a background image to your Stage.

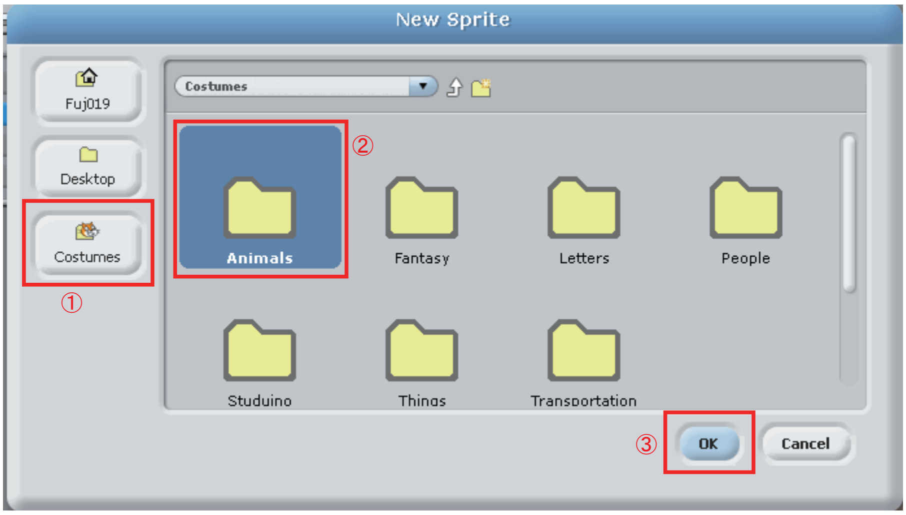


6.3.2.2. Adding Sprites

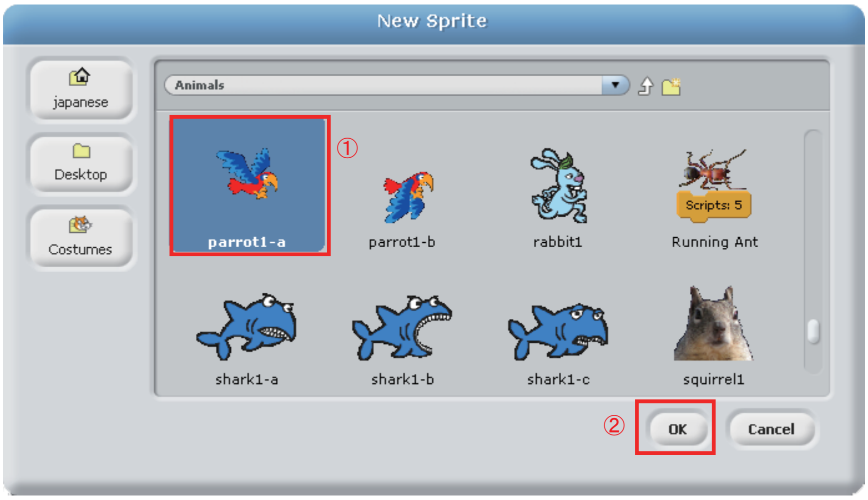
Find the **New sprite** bar under the Stage. Click the middle button.



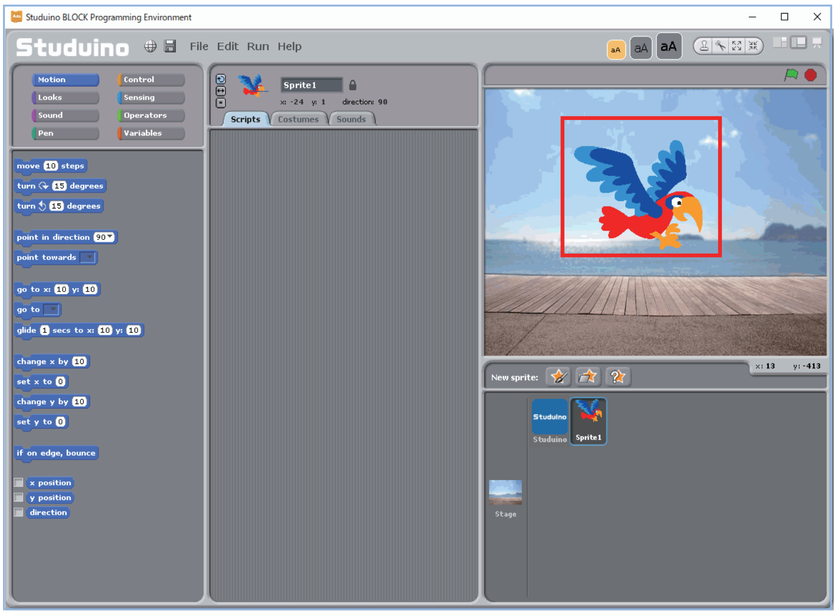
This will bring up the **New Sprite** window, giving you a selection of folders to choose from. Click the **Costumes** button (①), choose the **Animals** folder (②), and click the **OK** button (③).



In the **Import Costume** window that appears, choose **parrot1-a** (①) from the list of pictures and click the OK button (②).

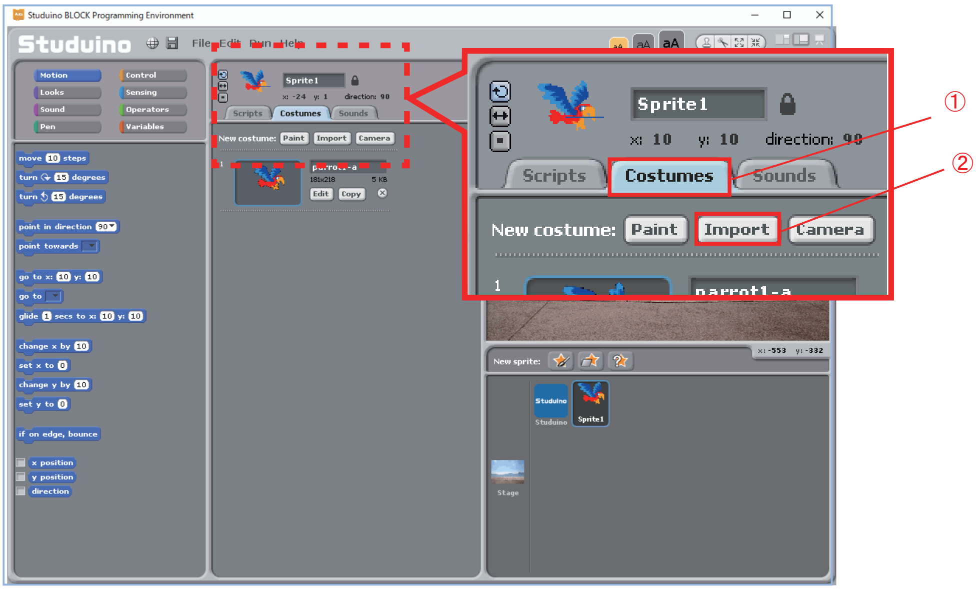


Now you've added a Sprite to your Stage.

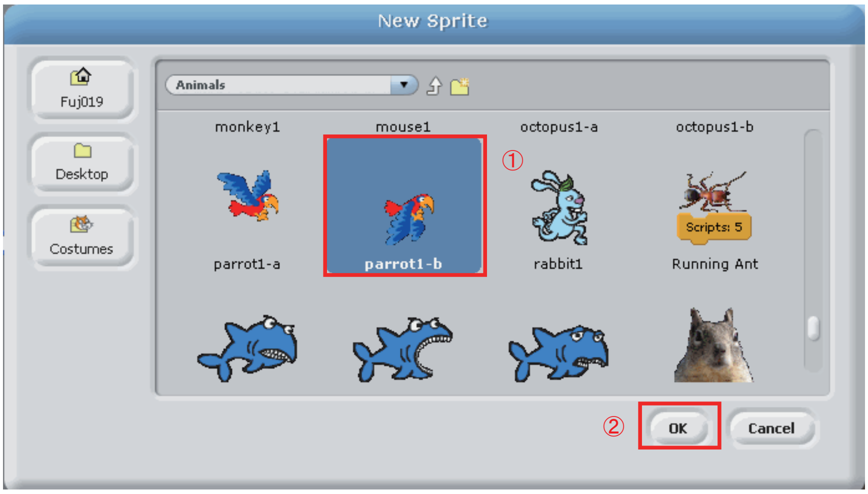


6.3.2.3. Adding a Sprite Costume

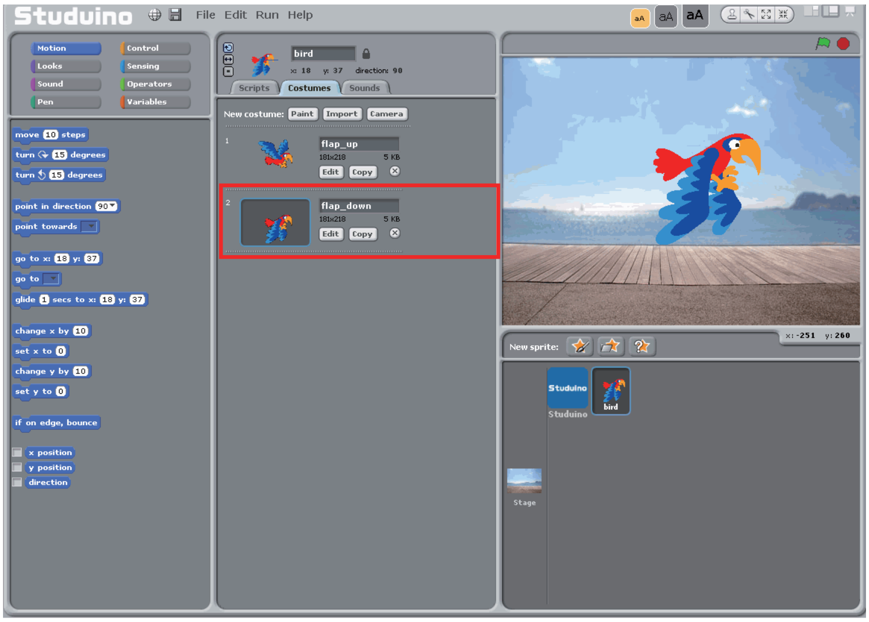
Click the **Costumes** tab (①) in the Script Field. Now click the **Import** button (②).



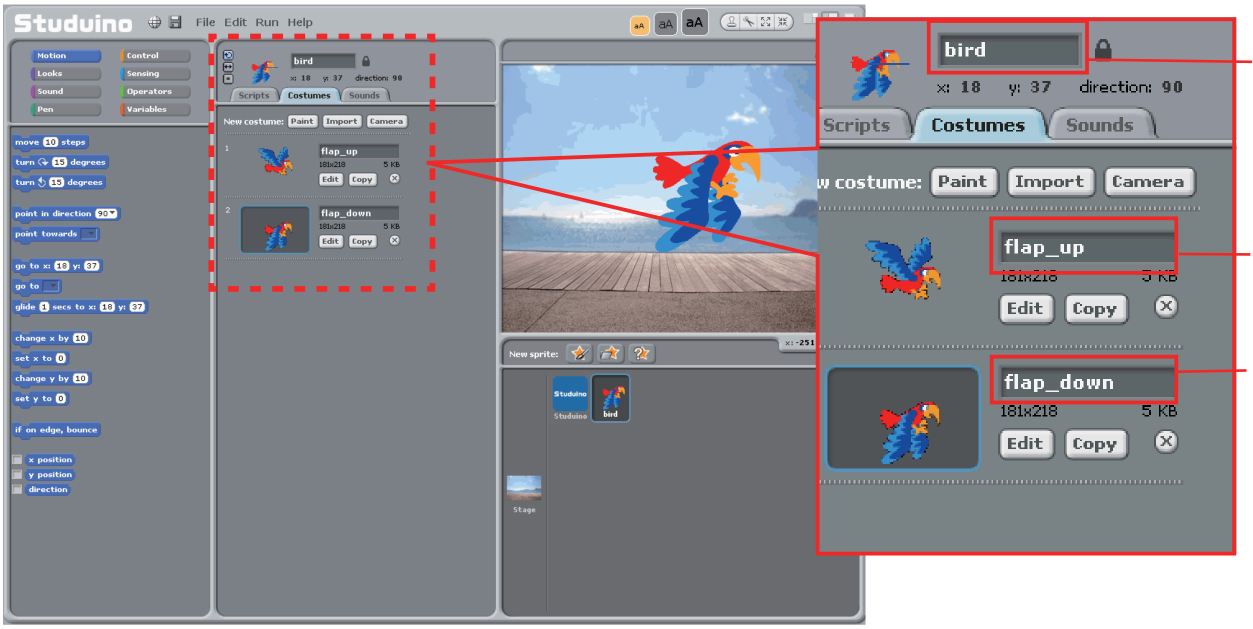
In the **Import Costume** window that appears, choose **parrot1-b** (①) from the list of pictures and click the **OK** button (②).



Now you’ve added a Costume to your Sprite.

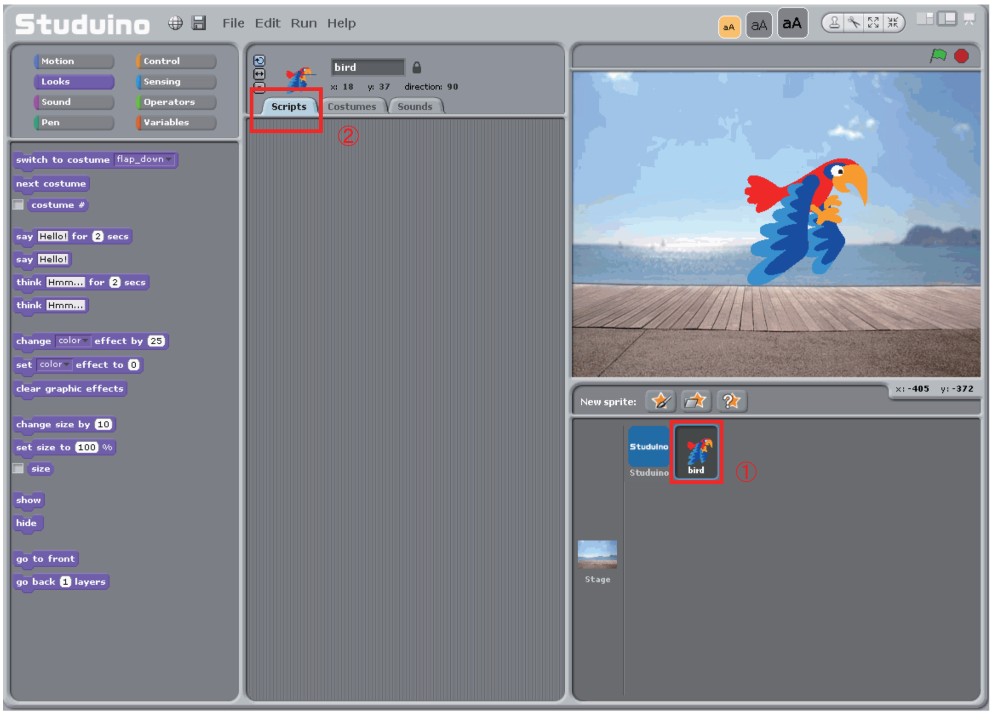


Name your Sprite **bird** (①) and give its Costumes the names **flap_up** (②) and **flap_down** (③) as shown below.

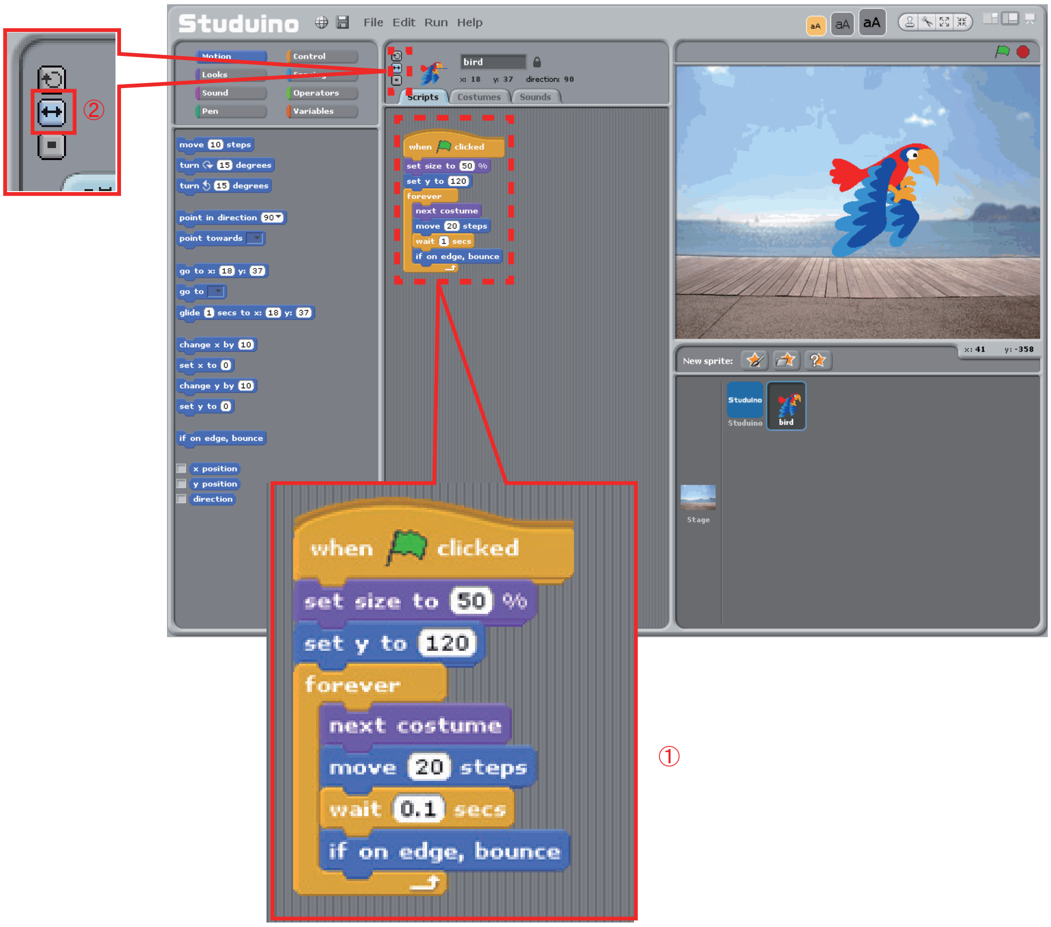


6.3.2.4. Programming Sprites

Click the **bird Sprite** (①). Now click the **Scripts** tab (②).

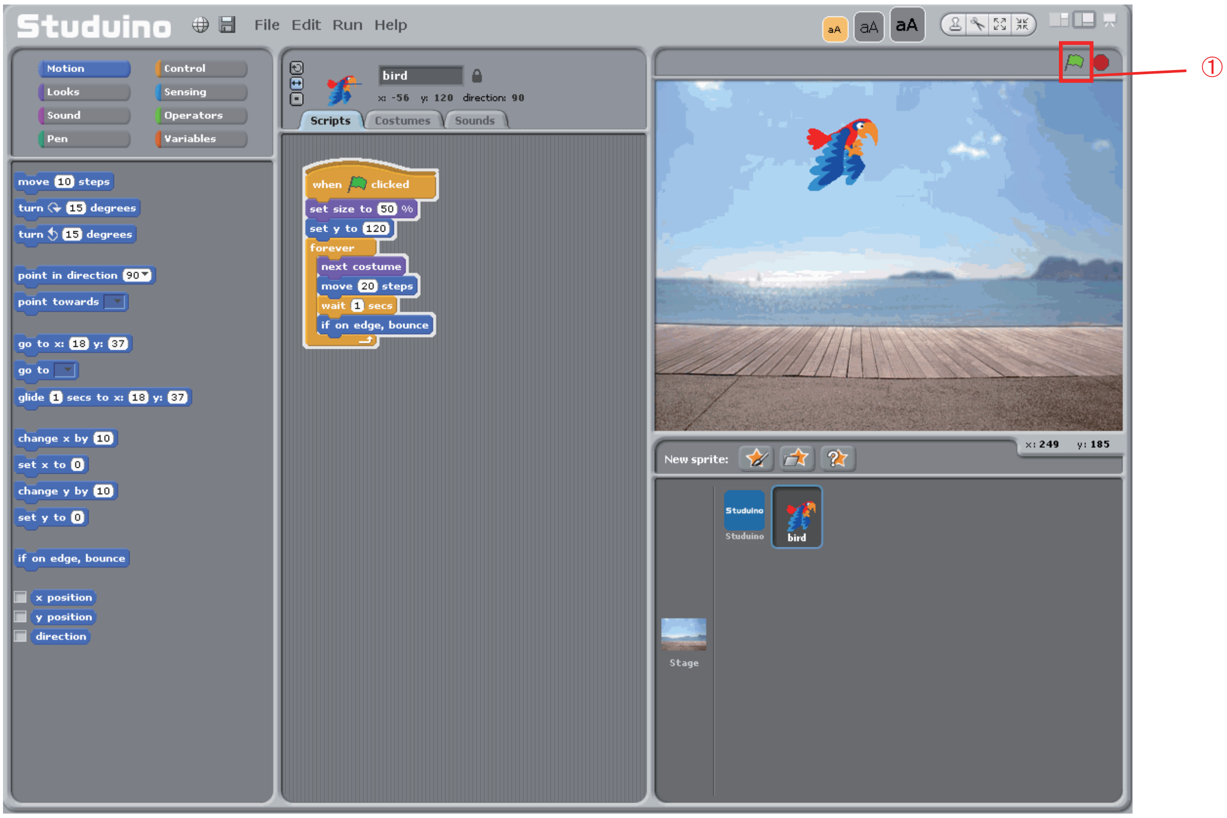


Now make the program shown below (①) in the Script Field. Click the **only face left-right** button (②).



```
when (green flag) clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

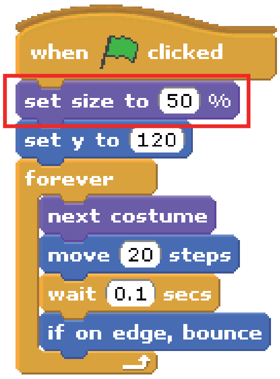
Click the **green flag** (①) as shown below and your bird will begin flying across the stage.



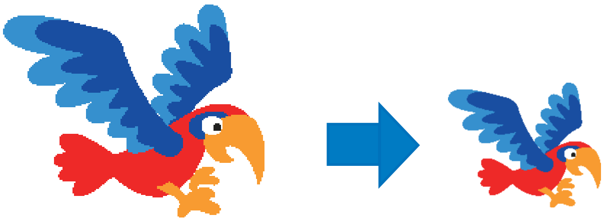
```
when [green flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 1 secs
    if on edge, bounce
```

Follow along below to see how the program works.

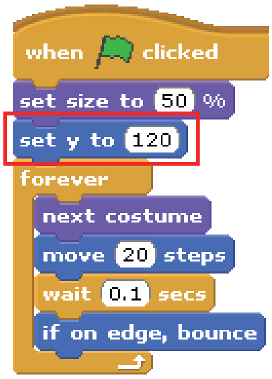
1. Click the green flag to run the set size to 50 % block. This block changes the size of a Sprite.



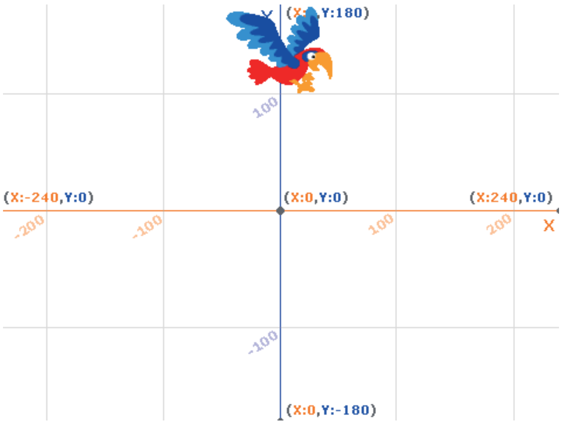
```
when [green flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```



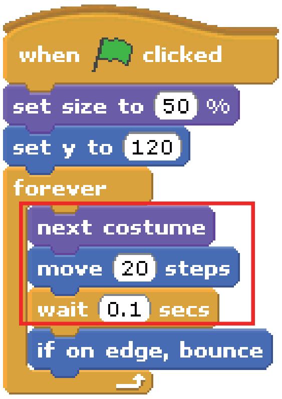
2. The set y to 120 block will run next. Every Stage has the coordinates shown below, and this block will make a Sprite move to y-coordinate 120.



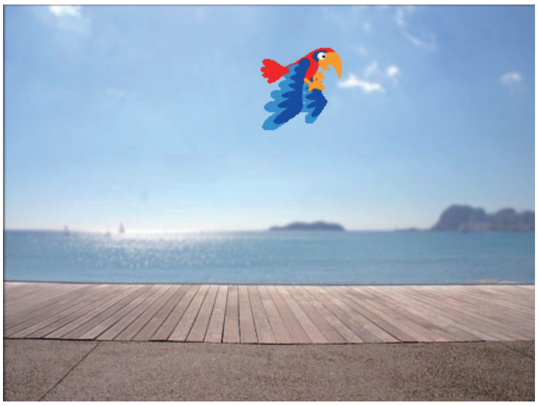
```
when [green flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```



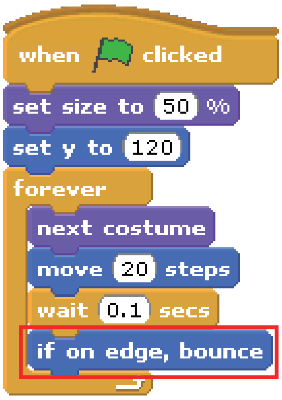
3. The next costume block will run next. This block sets the Sprite's Costume to **flap_down**. The move 20 steps block will run after this. This block makes the Sprite move for 20 steps. Lastly, the wait 0.1 secs block will run, making the program wait for **0.1** seconds.



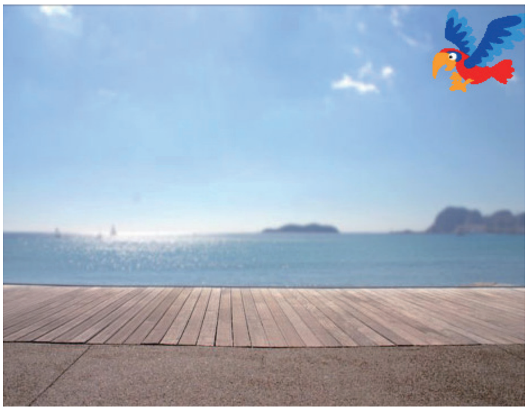
```
when [flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```



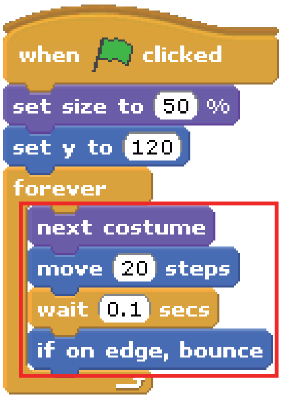
4. The if on edge, bounce block will make the Sprite turn around when it reaches the edge of the Stage.



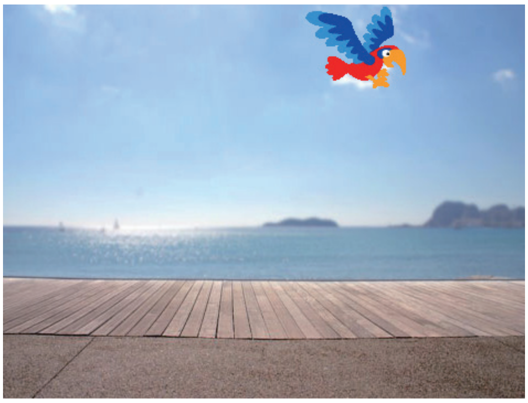
```
when [flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```



5. Steps 3 and 4 will repeat forever. Repeatedly changing your bird's Costume as it takes 20 steps makes an animation that looks like the bird is flying.



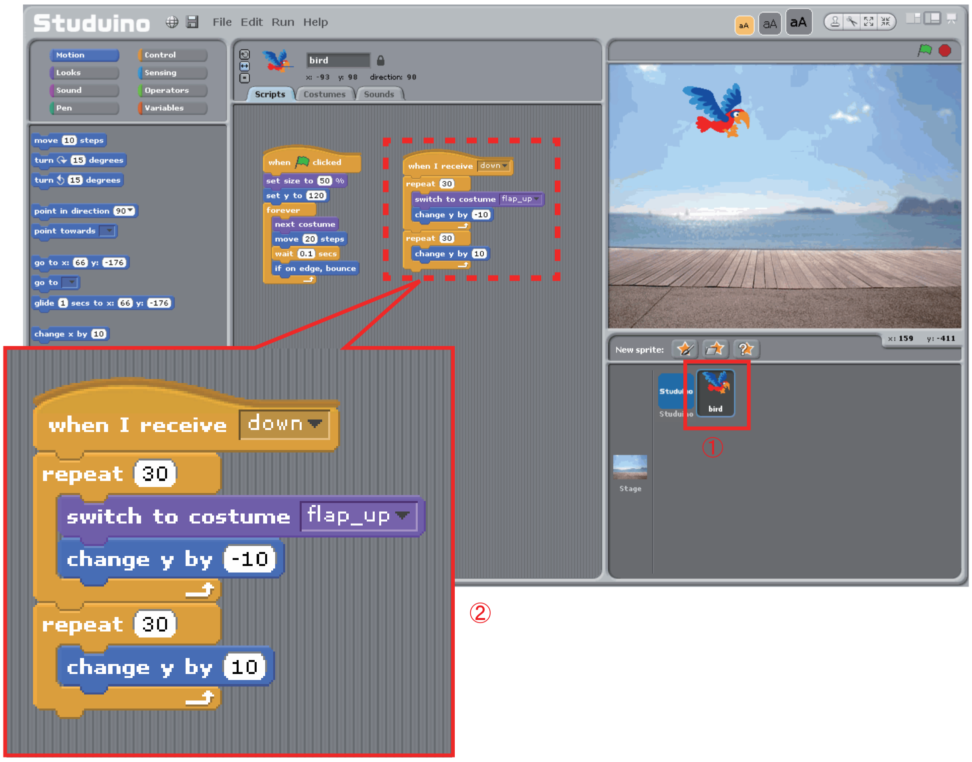
```
when [flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```



6.3.2.5. Exchanging Messages Between Sprites and Studuino

You can also make programs that link with your Studuino by using when I receive and broadcast blocks, allowing your Studuino and bird Sprites to send messages back and forth.

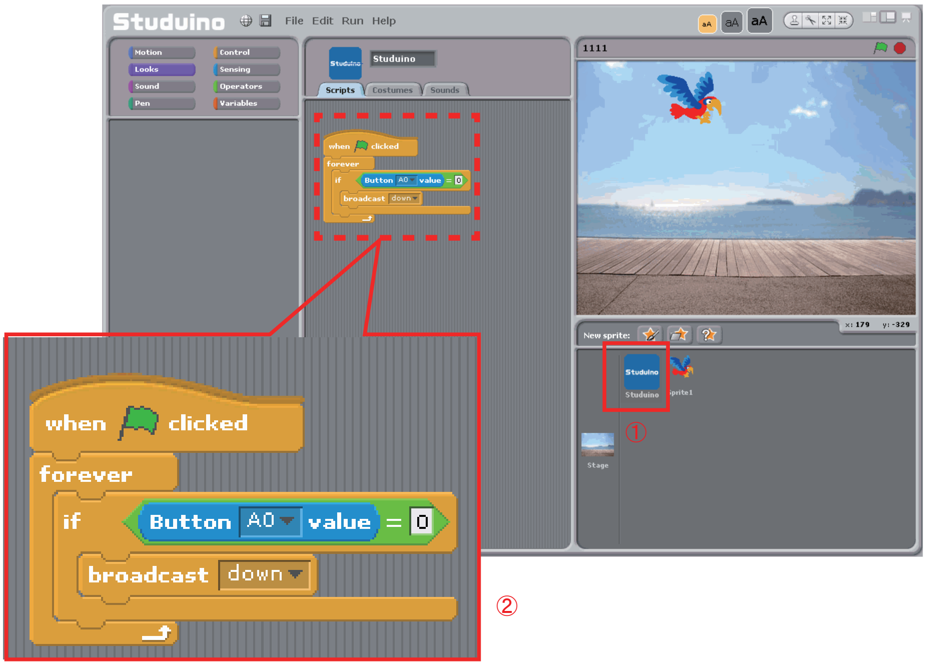
As shown below, click on the bird Sprite (①) in the Sprite List to add its program to the Script Field.



```
when [flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

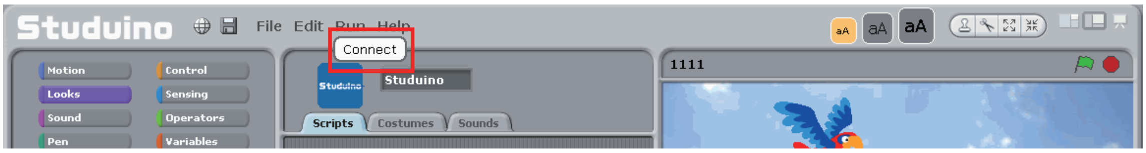
```
when I receive down
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

As shown below, click on the **Studuino** Sprite (①) in the Sprite List to add its program to the Script Field.

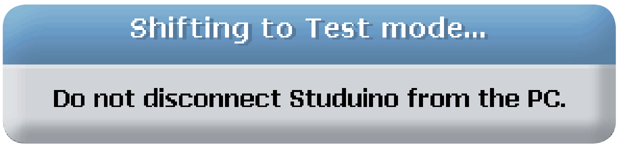


```
when [flag] clicked
forever
    if Button A0 value = 0
        broadcast down
```

Pick **Connect** from the **Run** menu.



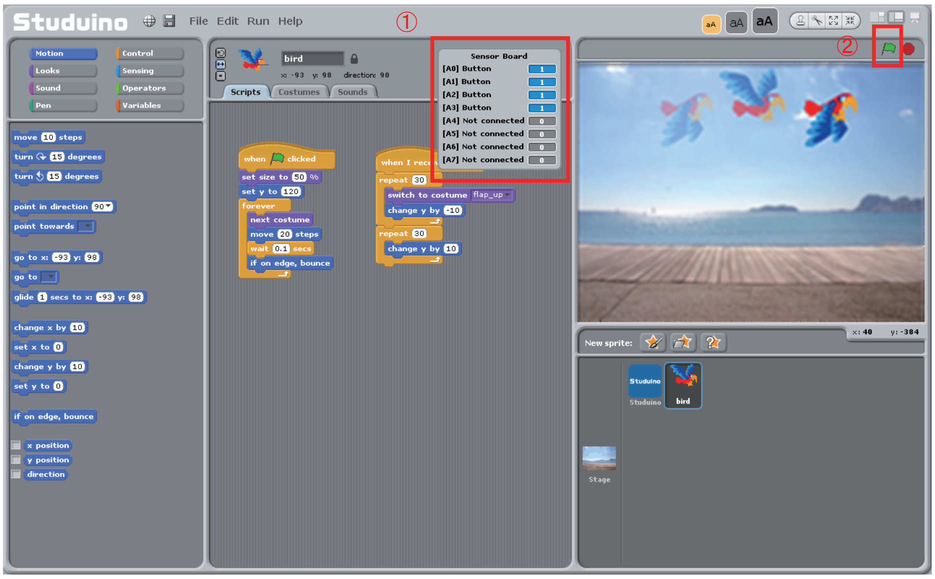
You'll see the message below as **Test Mode** opens.



Shifting to Test mode...

Do not disconnect Studuino from the PC.

The message will disappear and the **Sensor Board** (①) will appear. Click the **green flag** (②) and your bird will fly around the Stage.

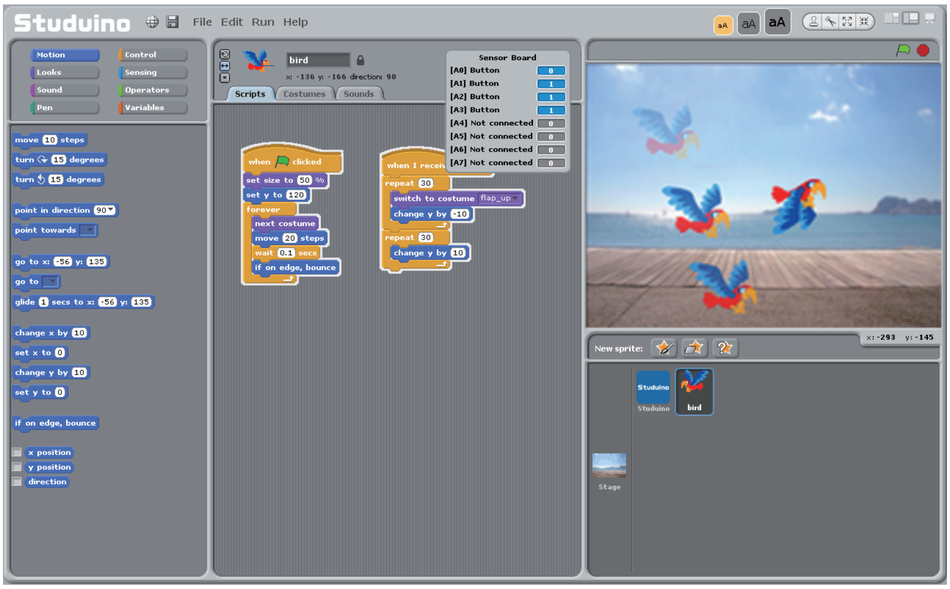


Sensor Board	
[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] Button	1
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

```
when 🏁 clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

```
when I rece
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

Press **A0** on your Studuino and the bird will land before taking off again.

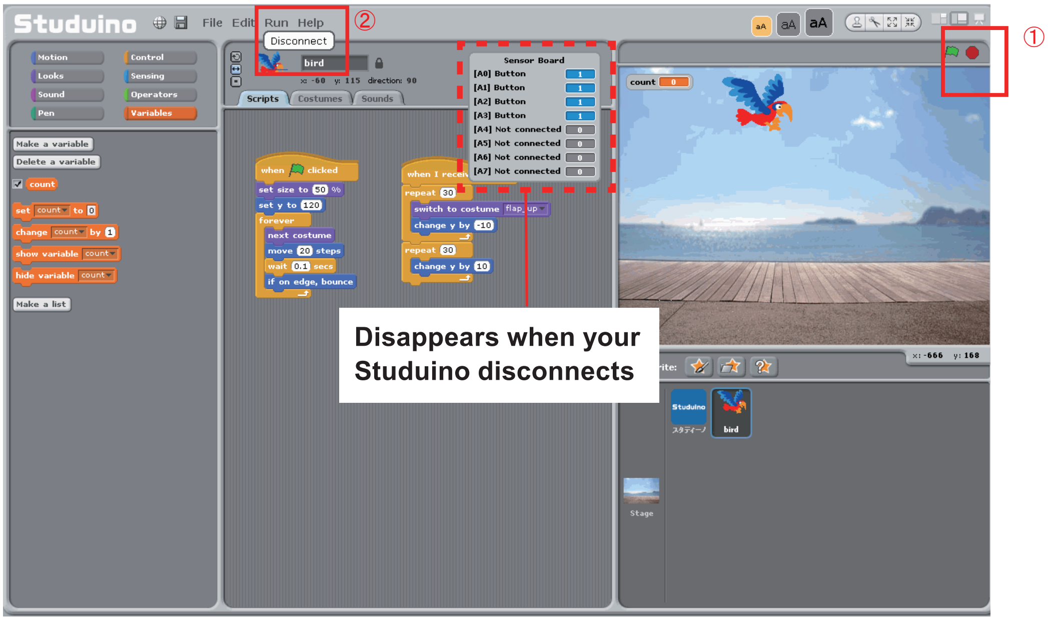


Sensor Board	
[A0] Button	0
[A1] Button	1
[A2] Button	1
[A3] Button	1
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

```
when 🏁 clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

```
when I recei
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

Press the **Stop** button (①) as shown below to stop your bird. Choose **Disconnect** (②) from the **Run** menu to disconnect your Studuino from your PC. The Sensor Board will disappear.



```
when [flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

```
when I receiv
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

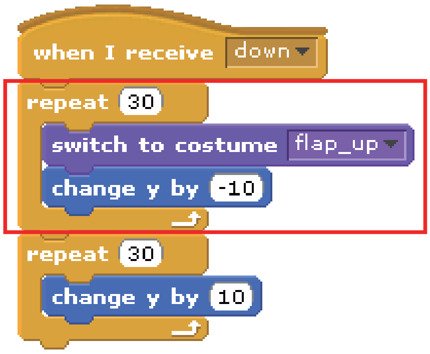
Sensor Board

[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] Button	1
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

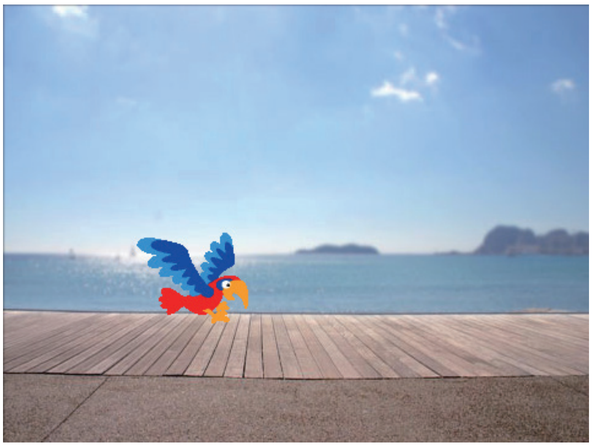
Disappears when your Studuino disconnects

Follow along below to see how the program works.

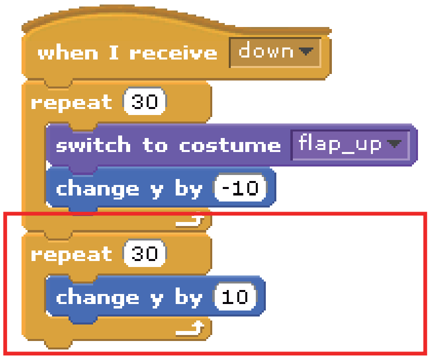
1. Your bird Sprite will change to its **flap_down** Costume when it receives the **down** command, repeating a -10 point step down the Y-axis 30 times.



```
when I receive down
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

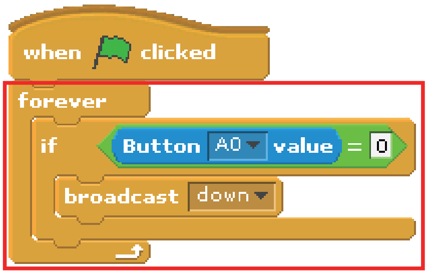


2. It will then repeat a 10 point step up the Y-axis 30 times to return to its original height.

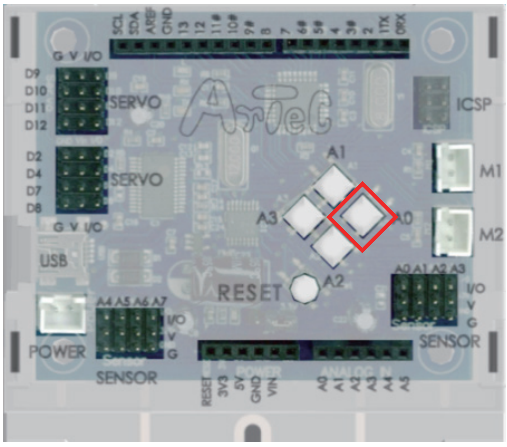


```
when I receive down
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

The Studuino Sprite will check whether or not the value of **A0** is **0** (this means that you've pressed the button) forever. It sends the down command when you press **A0**.



```
when clicked
forever
    if Button A0 value = 0
        broadcast down
```

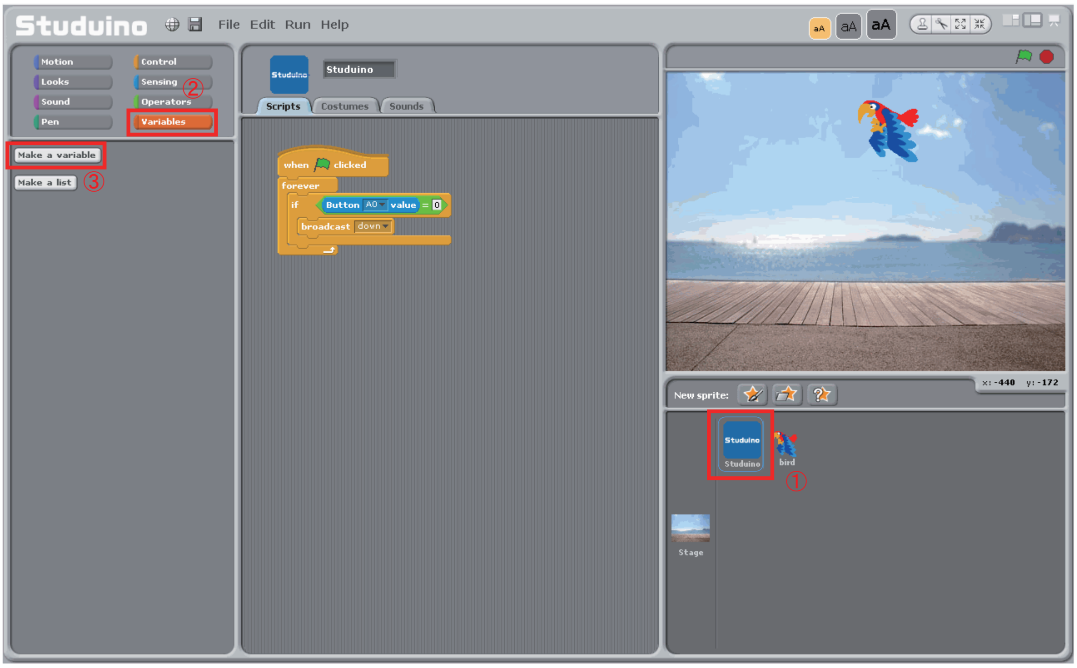


Studuino

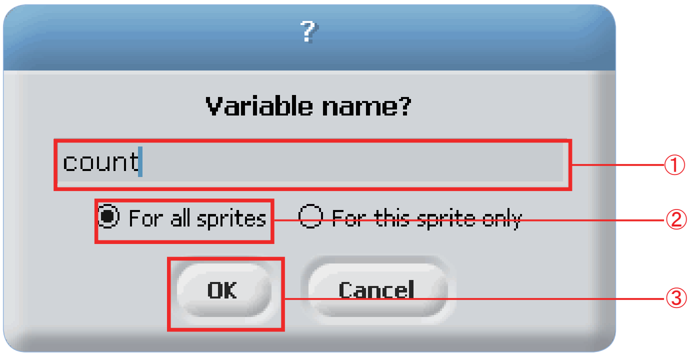
In this example the **Studuino** Sprite sends the **down** command when you press **A0**, and the **bird** Sprite lands and takes off when it receives the message. This means that the bird will land and take off each time you press the button. This repeated exchange of messages allows your Studuino to control the Sprites.

6.3.2.6. Controlling Sprites with Variables

You can also use the **Studuino** Sprite to make variables. Click the **Studuino** Sprite (①). Now click the **Variables** button (②) and click the **Make a new variable** button (③).



Name the variable **count** (①) and choose **For all sprites** (②). Now click **OK** (③).


?

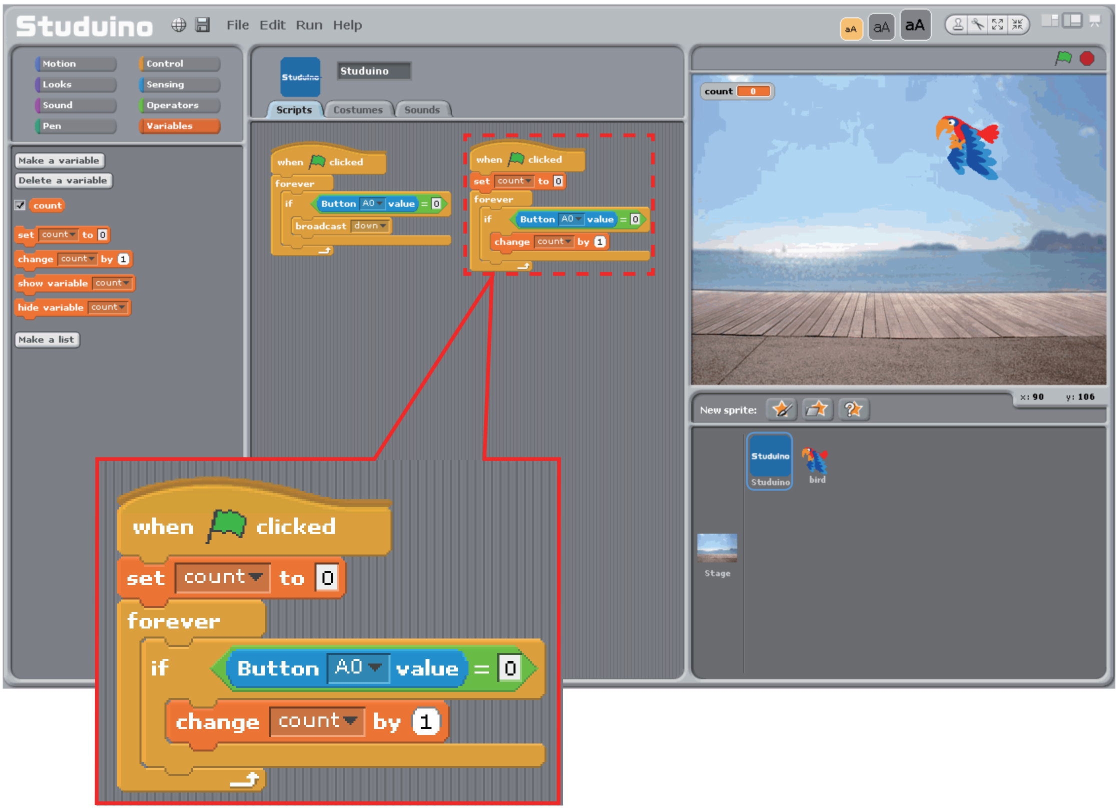
Variable name?

count

● For all sprites ○ For this sprite only

OK Cancel

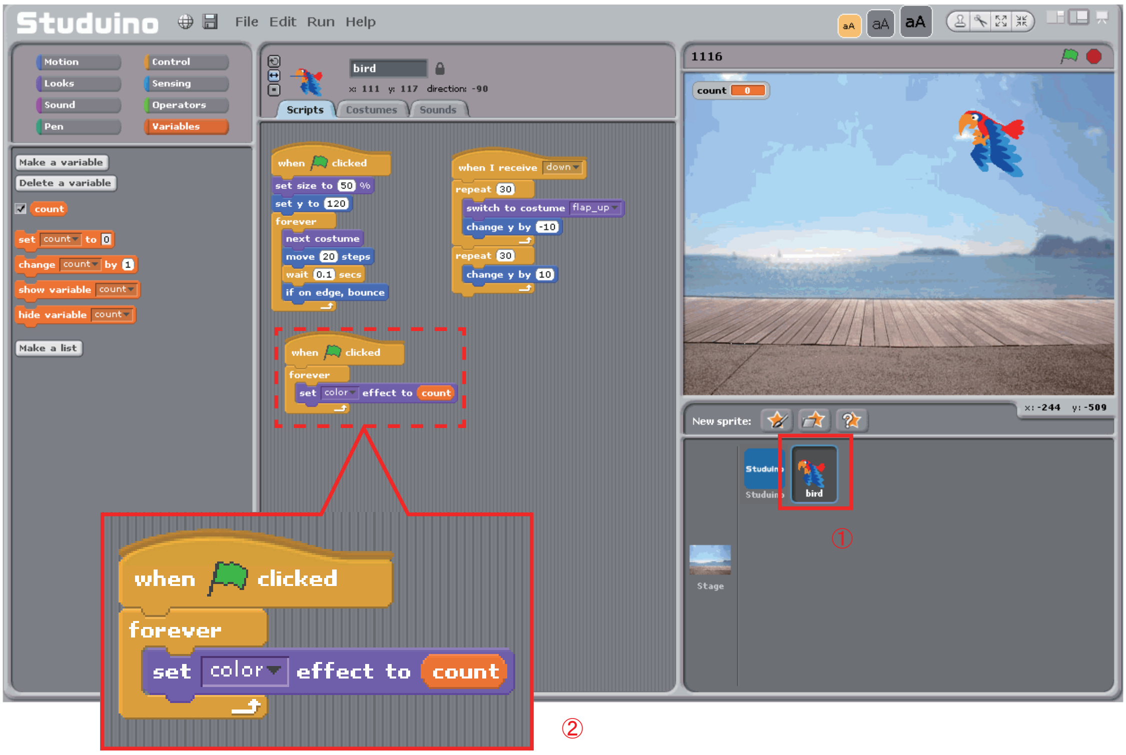
Now add the program shown below to the Script Field.



```
when [flag] clicked
forever
  if Button A0 value = 0
    broadcast down
```

```
when [flag] clicked
set count to 0
forever
  if Button A0 value = 0
    change count by 1
```

As shown below, click the **bird** Sprite (①) in the Sprite List to add this program to the Script Field.

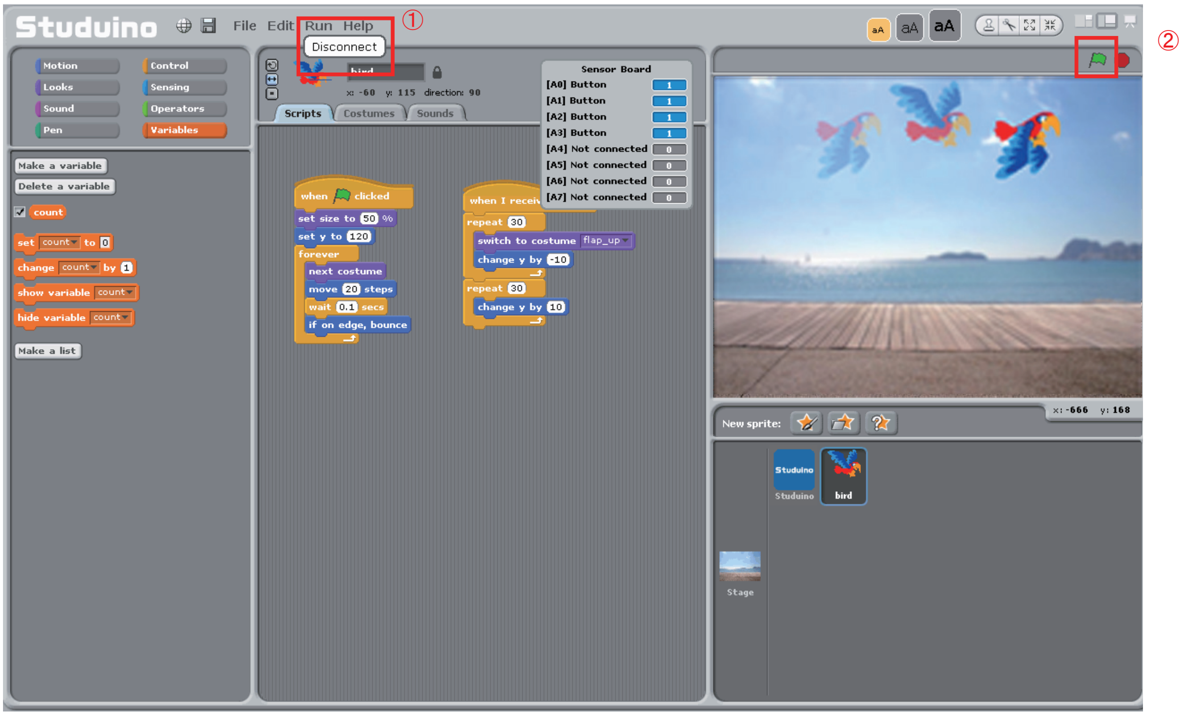


```
when [flag] clicked
set size to 50 %
set y to 120
forever
  next costume
  move 20 steps
  wait 0.1 secs
  if on edge, bounce
```

```
when I receive down
repeat 30
  switch to costume flap_up
  change y by -10
repeat 30
  change y by 10
```

```
when [flag] clicked
forever
  set color effect to count
```

Use a USB cable to connect your Studuino and PC and choose **Connect** (①) from the **Run** menu. Once they're connected, click the **green flag** (②) to make your bird fly around the Stage.

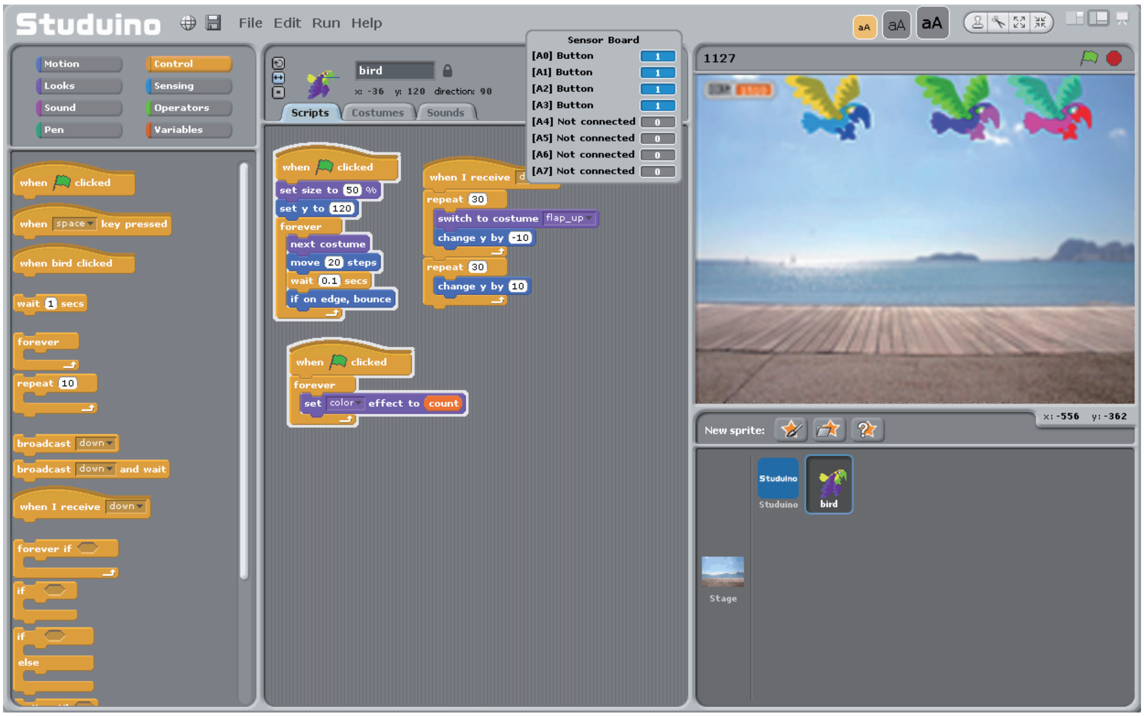


```
when [green flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

```
when I receiv
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

Sensor Board	
[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] Button	1
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

Press **A1** on your Studuino and the bird will change color.



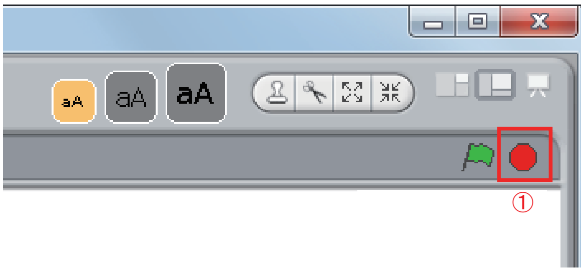
```
when [green flag] clicked
set size to 50 %
set y to 120
forever
    next costume
    move 20 steps
    wait 0.1 secs
    if on edge, bounce
```

```
when I receive d
repeat 30
    switch to costume flap_up
    change y by -10
repeat 30
    change y by 10
```

```
when [green flag] clicked
forever
    set color effect to count
```

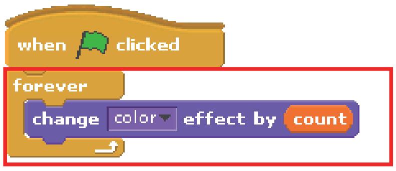
Sensor Board	
[A0] Button	1
[A1] Button	1
[A2] Button	1
[A3] Button	1
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

Stop your program by clicking the **Stop** button (figure ① on the right). Now disconnect your Studuino and PC by choosing **Disconnect** from the **Run** menu. The Sensor Board will disappear once they've disconnected.



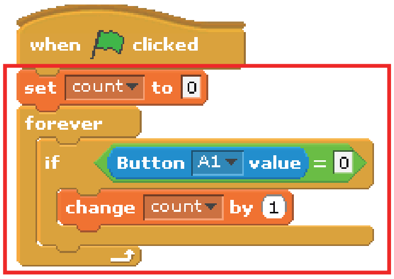
Follow along below to see how the program you've added works.

Your **bird** Sprite will change its color based on the value in the **count** variable.

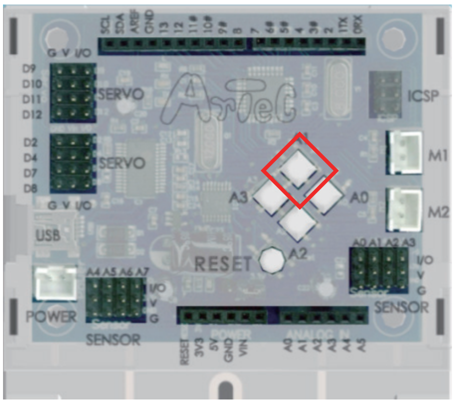


```
when 🏴 clicked
forever
    change color effect by count
```

Once the **count** variable is **0**, the **Studuino** Sprite will check whether or not the value of **A1** is **0** (this means that you've pressed the button) **forever**. Pressing **A1** will add **1** to the value of the **count** variable.



```
when 🏴 clicked
set count to 0
forever
    if Button A1 value = 0
        change count by 1
```



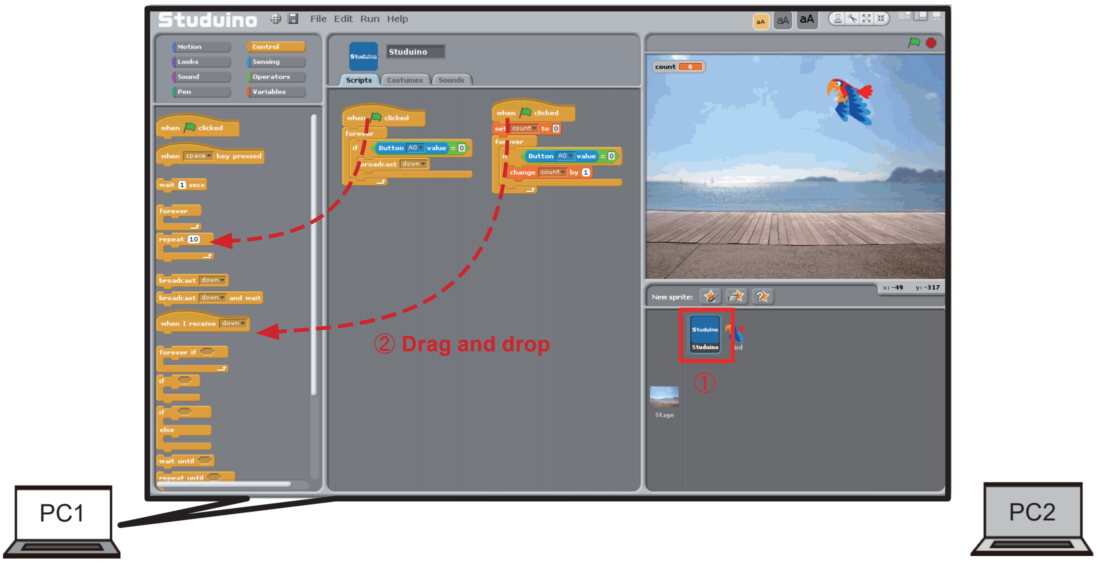
Studuino

Pressing **A1** will make the **Studuino** Sprite update the **count** variable, and your **bird** Sprite will use this variable to change colors every time you press the button. Using variables in this way allows your Studuino to control Sprites.

6.3.2.7. Exchanging Messages Through a Mesh

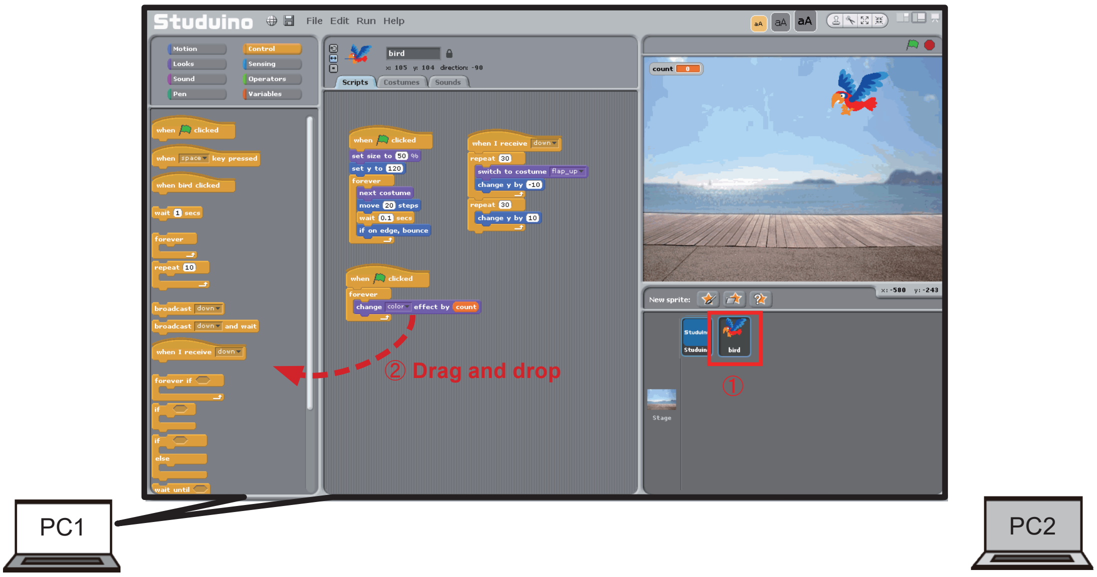
This software can use Mesh networks to link multiple PCs. Follow the steps below to use this software on two PCs at once. The PCs you'll use will need to be connected through an Ethernet cable or a WiFi network. **PC1** will be the PC you used from **6.3.2.1. Adding a Stage** to **6.3.2.6. Controlling Sprites with Variables**, while the PC you'll be setting up in this section will be **PC2**.

Delete your Studuino Sprite program from **PC1**. As shown below, click the **Studuino** Sprite (①) and drag the when clicked blocks (②) back to the Command Palette to delete them.



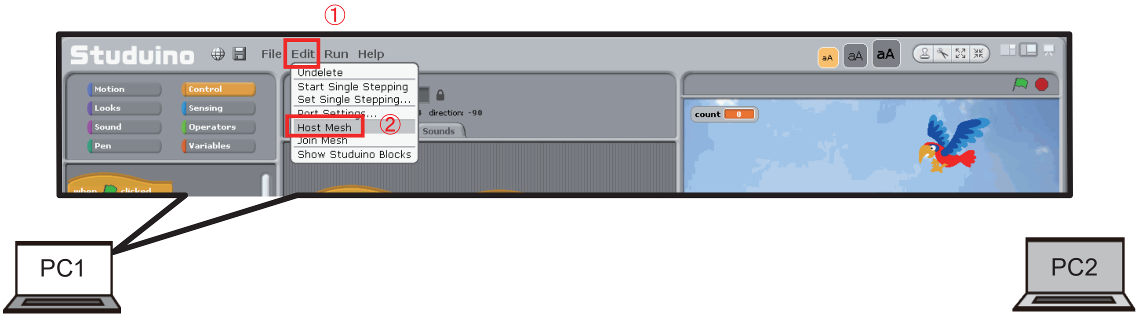
```
when clicked
forever
  if Button A0 value = 0
    broadcast down

when clicked
set count to 0
forever
  if Button A0 value = 0
    change count by 1
```

You'll also need to click the **bird** Sprite (①) and drag the count block (②) back to the Command Palette to delete it.

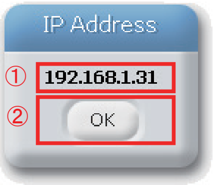


```
when clicked
set size to 50 %
set y to 120
forever
  next costume
  move 20 steps
  wait 0.1 secs
  if on edge, bounce

when I receive down
repeat 30
  switch to costume flap_up
  change y by -10
repeat 30
  change y by 10

when clicked
forever
  change color effect by count
```

Next you'll need to setup your network. Go to the **Edit** menu (①) on **PC1** and click **Host Mesh** (②).



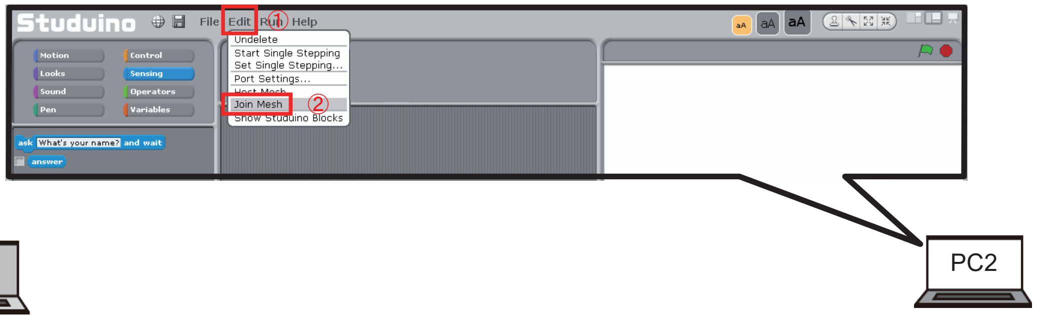
Write down the IP address (①) and click **OK** (②).



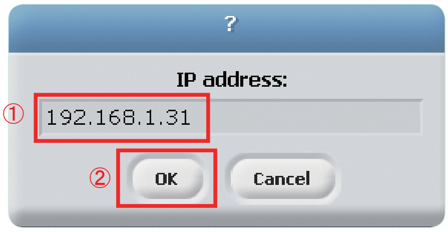
IP Address

192.168.1.31

Go to the **Edit** menu (①) on PC2 and click **Join Mesh** (②).



Enter the IP address (①) and click **OK** (②).

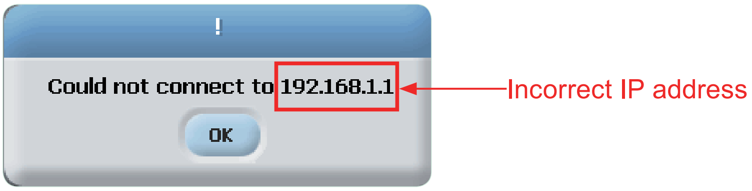


?

IP address:

192.168.1.31

Enter an incorrect IP address and you'll see the message below.

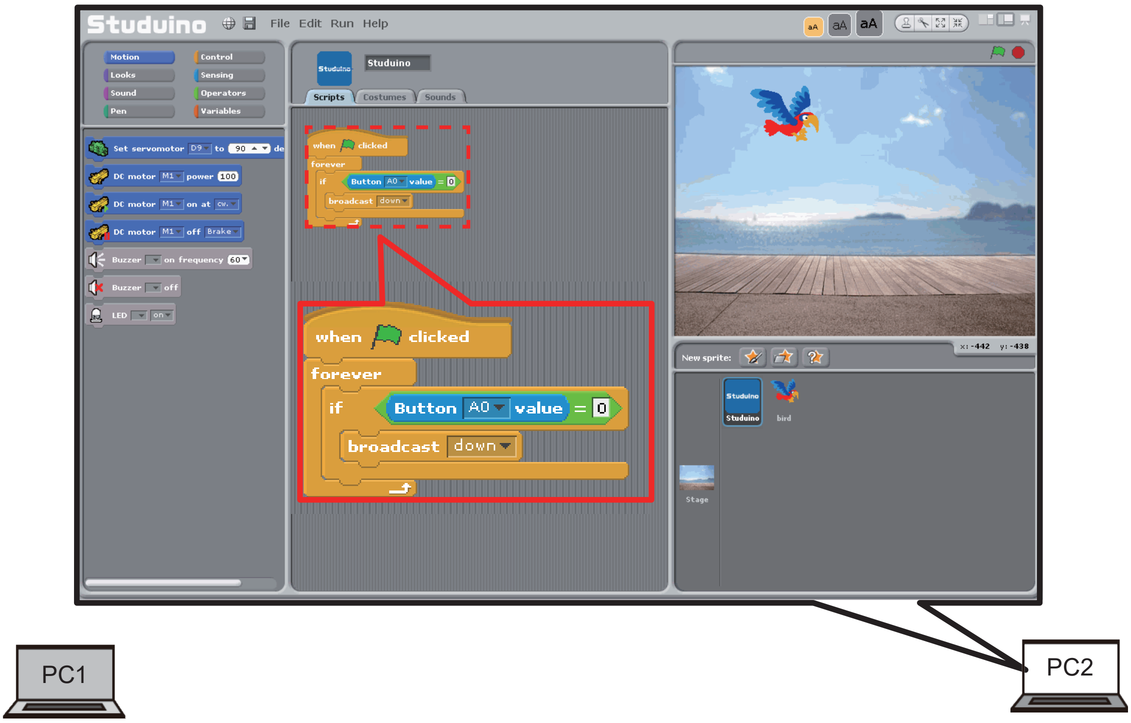


!

Could not connect to 192.168.1.1

PC2 has successfully joined the Mesh if you don't see this message.

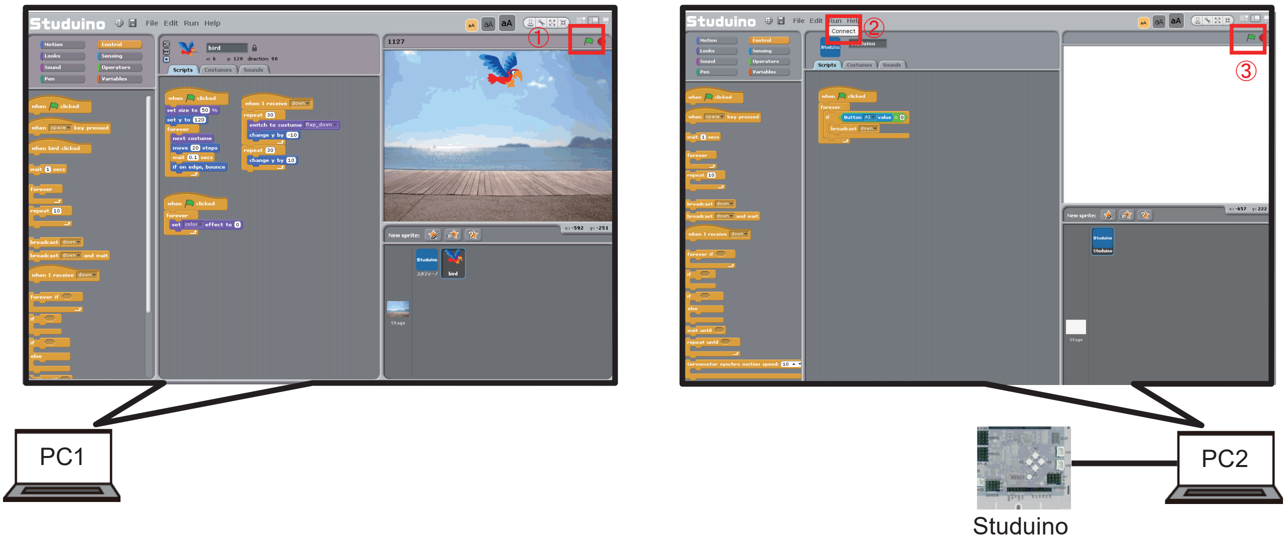
Sending messages between each PC allows your Studuino to link with Sprites across the Mesh. We’re going to make the program shown below (①) in the Script Field of PC2.



```
when [flag] clicked
forever
  if < Button A0 value = 0 >
    broadcast down
```

Follow the steps below to run your program:

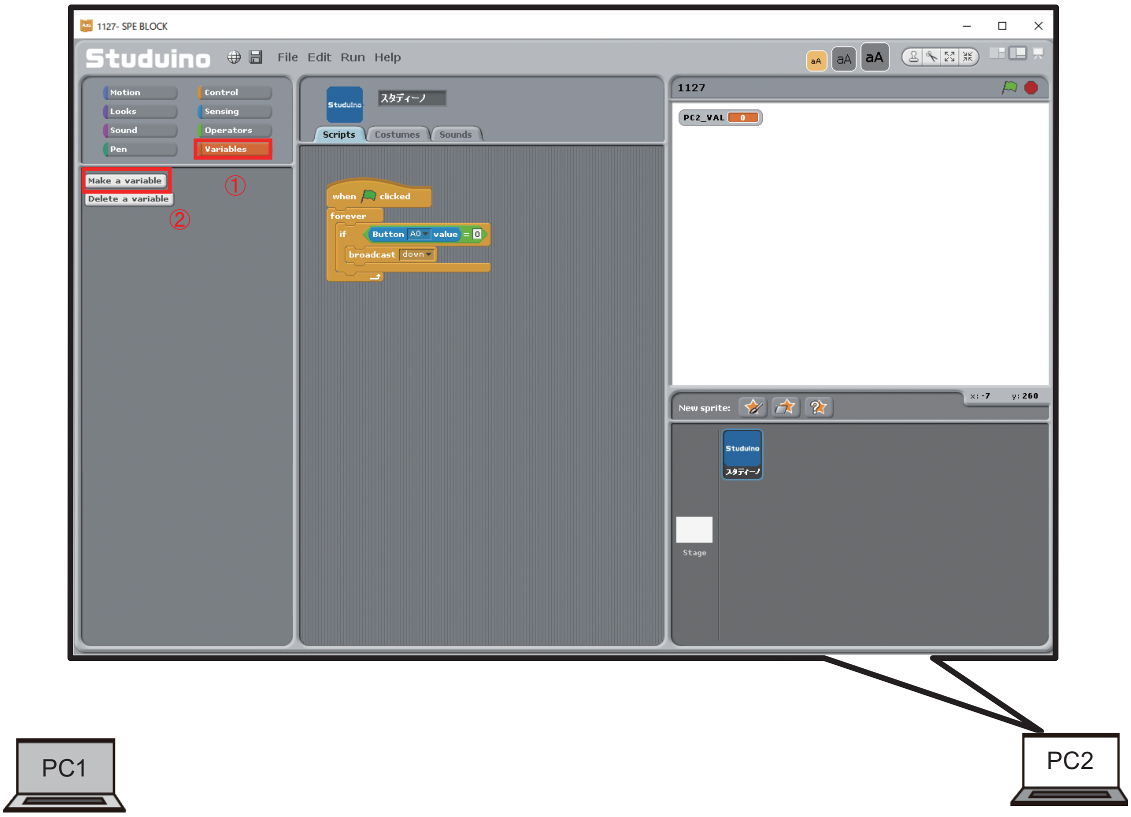
1. Click the **green flag** (①) on **PC1** and your bird will begin flying across the stage.
2. Use a USB cable to connect your Studuino and **PC2**.
3. Now choose **Connect** from the **Run** menu on **PC2**.
4. Click the **green flag** (③) once your Studuino and **PC2** are connected.



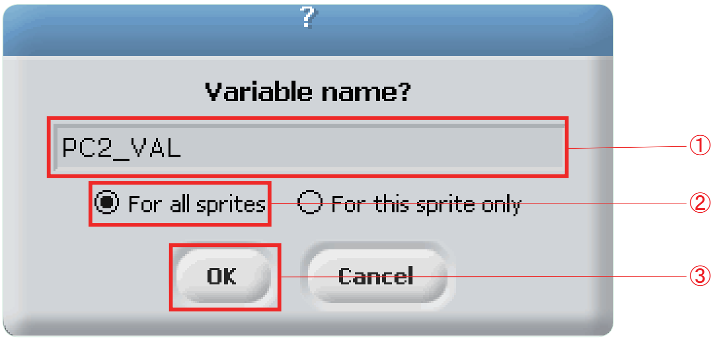
Press **A0** on your Studuino and see if the bird on **PC1** will land before taking off again. Now you’ve used a Studuino connected to **PC2** to control the bird on **PC1**. This is how using Meshes and messages allows you to use a Studuino to control Sprites over a network.

6.3.2.8. Sharing Variables on a Mesh

You can also link your Studuino to Sprites over a network by sharing variables between your PCs. You'll need to use your Studuino Sprite on **PC2** to make a variable. Click the **Variable** button (①). Now click **Make a variable** (②).



Name the variable **PC2_VAL** (①) and choose **For all sprites** (②). Now click **OK** (③).

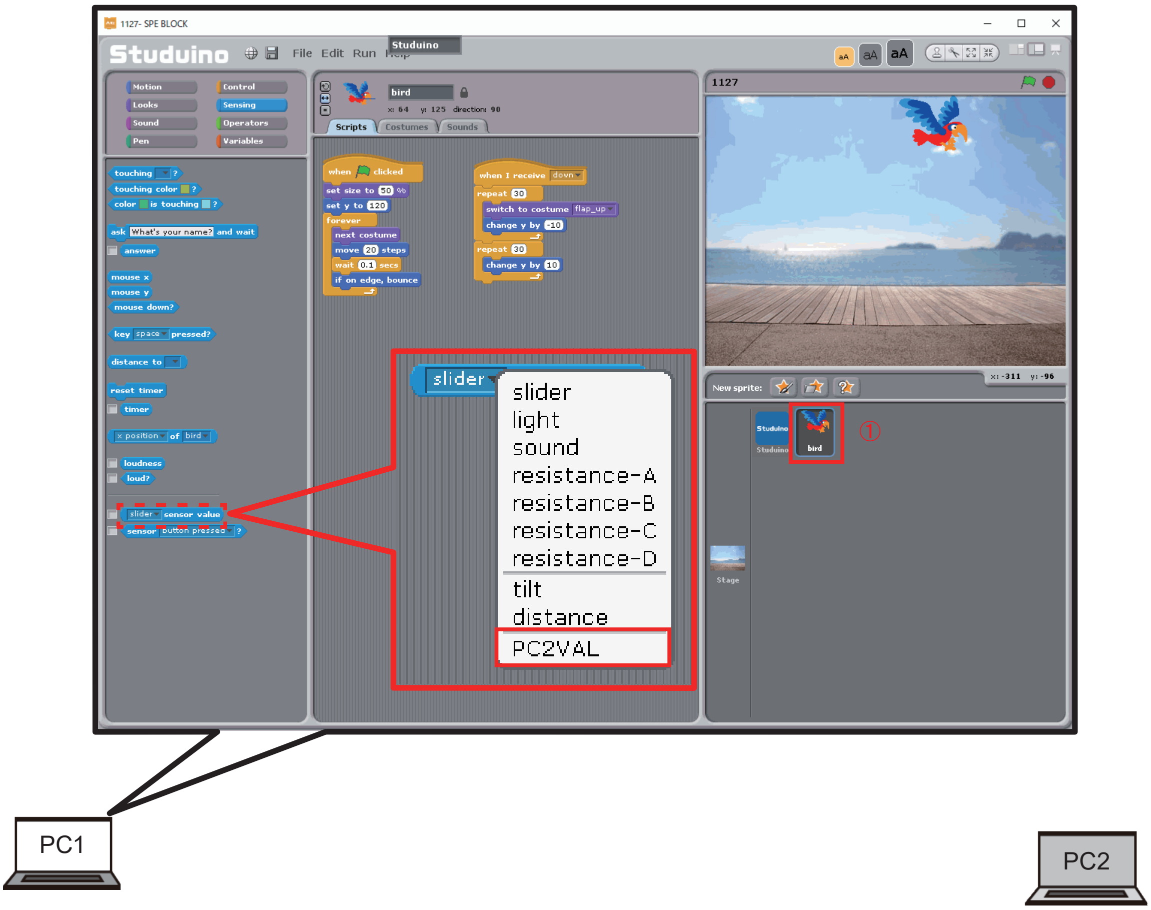


Variable name?

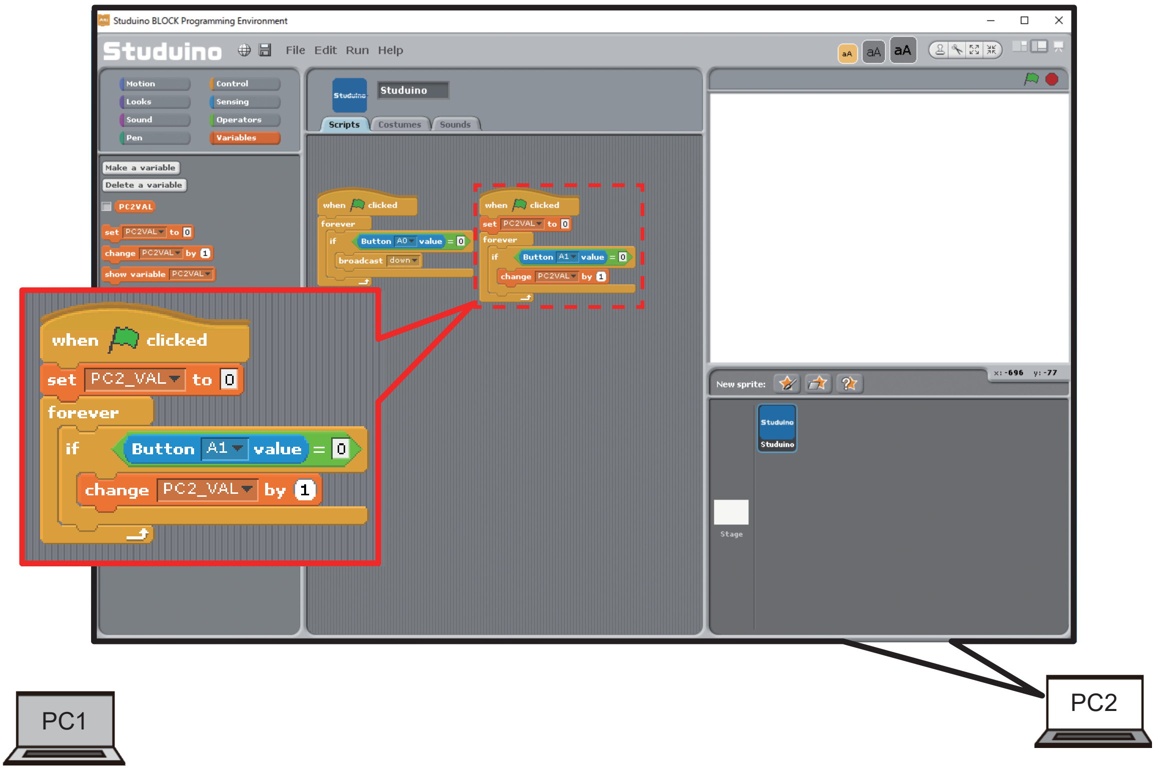
PC2_VAL

● For all sprites ○ For this sprite only

You'll be able to see the value of the variable on **PC2** by using the **bird** Sprite's `slider▾ sensor value` block on PC1. Click the **bird** Sprite on **PC1**. Now click the ▼ in the `slider▾ sensor value` block and choose the **PC2_VAL** variable you made on **PC2**.

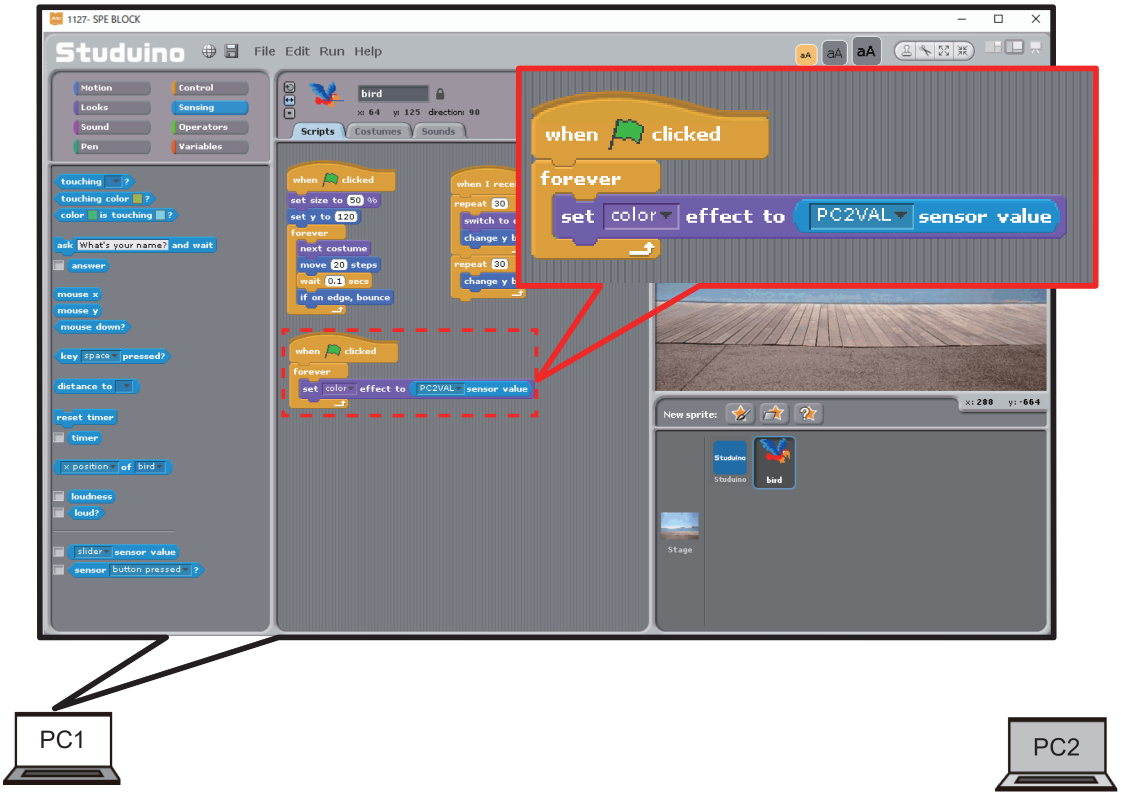


Now add the program below to the Script Field of **PC2**.



```
when 🏴 clicked
set PC2_VAL▾ to 0
forever
  if Button A1▾ value = 0
    change PC2_VAL▾ by 1
```

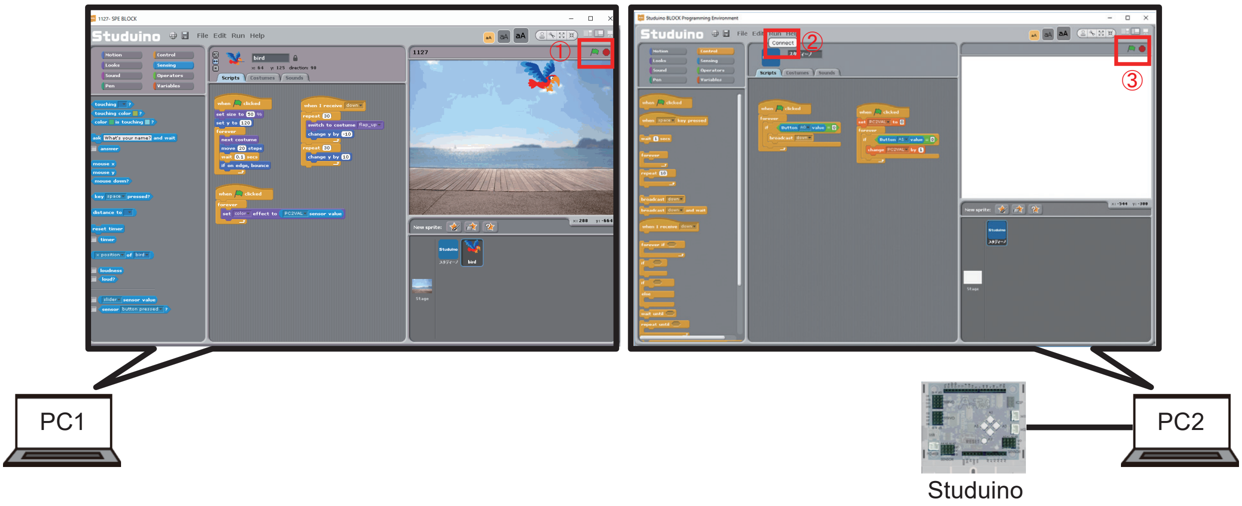
Now add the program below to the Script Field of **PC1**.



```
when 🏴 clicked
forever
    set color effect to PC2VAL sensor value
```

Follow the steps below to run your program:

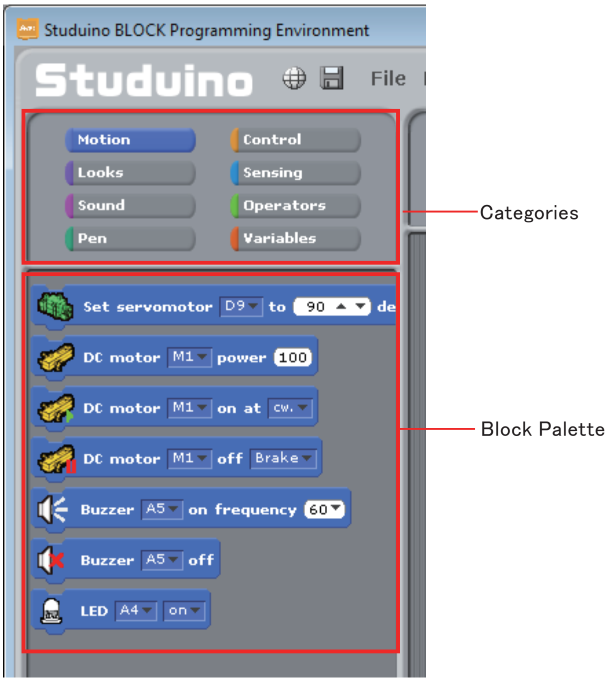
1. Click the **green flag** (①) on **PC1** and your bird will begin flying across the stage.
2. Use a USB cable to connect your Studuino and **PC2**.
3. Now choose **Connect** from the **Run** menu on **PC2**.
4. Click the **green flag** (③) once your Studuino and **PC2** are connected.



Press **A1** on your Studuino and the bird on **PC1** will change color. Now you've used a Studuino connected to **PC2** to control the bird on **PC1**. This is how using Meshes and variables allows you to use a Studuino to control Sprites over a network.

6.3.3. Studuino Sprite Blocks

You'll need to use the blocks in the Block Palette when you want to program Stages, Sprites, or the Studuino Sprite. The blocks you'll use to program are divided into **Motion**, **Control**, **Looks**, **Sensing**, **Sound**, **Operators**, **Pen**, and **Variables**. You can select a category of blocks by click on its button at the top of the Command Palette. Look below to see a description of the blocks you'll use with your Studuino Sprite. Read Scratch documentation or visit the Scratch Wiki (1) to learn more about the Stage and Sprite blocks.



Category	What the Blocks Do
Motion	Control DC Motors, Servomotors, Buzzers, and LEDs
Control	Control basic programming elements such as conditions, functions, and wait processes
Sensing	Lookup sensor values
Operators	Control basic and advanced arithmetic and logic operators
Variables	Make and control variables and lists
Looks	Looks, Sound, and Pen blocks can't be used with the Studuino Sprite.
Sound	
Pen	

[1] https://wiki.scratch.mit.edu/wiki/Blocks_(1.4)

6.3.4. More About Blocks

See **6.2.2.1. More About Blocks** to learn more about programming with blocks.

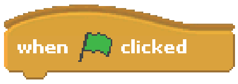
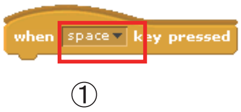
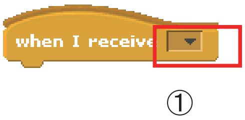
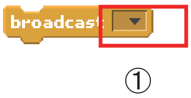
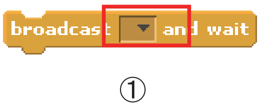
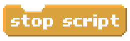
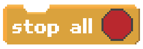
The following section explains the different types of blocks in the Block Palette.

- **Motion Palette Blocks**

Blocks in the Motion Palette are used to control the parts of your robot. You can find out more about these blocks in 6.2.2.1. Motion Palette Blocks.

- **Control Palette Blocks**

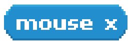
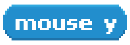
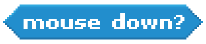
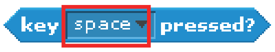
Blocks in the Control Palette allow you to control the flow of your program. Find descriptions of each block below.

Block	Description
	Click the green flag to run any blocks connected to this one.
	Press the key you set in ① to run any blocks connected to this one.
	Receiving the message set in ① will run any blocks connected to this one.
	Sends the message set in ① .
	Sends the message set in ① and waits for the message to be received.
	Stops the blocks connected to it from running.
	Stops all blocks from running.

You can find out more about other blocks in this Palette in **6.2.2.1. Control Palette Blocks**.

■ Sensing Palette Blocks

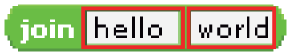
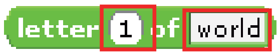
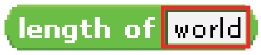
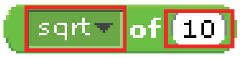
Blocks in the Sensing Palette are used to retrieve sensor values and external input from a keyboard. Find descriptions of each block below.

Block	Description
	Detects mouse movement along the Stage's X-axis.
	Detects mouse movement along the Stage's Y-axis.
	Detects whether the left mouse button has been clicked. It will be true when the button is clicked and false when it isn't.
 ①	Detects whether the key set in ① has been pressed. It will be true when the key is pressed and false when it isn't.

You can find out more about other blocks in this Palette in **6.2.2.1. Sensing Palette Blocks**.

■ Operators Palette Blocks

The Operators Palette contains blocks that will perform mathematical operations on the values you input. Find descriptions of each block below.

Block	Description
 ① ②	Joins the text strings you input in ① and ② . This block would return the string "hello world".
 ② ①	Returns the position set in ② of the text string set in ① . This block would return the string "w".
 ①	Looks up the number of characters of the text string set in ① . This block would return 4.
 ② ①	Performs the operation in ② on the value set in ① . You can choose from absolute values, square roots, trigonometric functions, logarithms, and exponents.

You can find out more about other blocks in this Palette in **6.2.2.1. Operators Palette Blocks**.

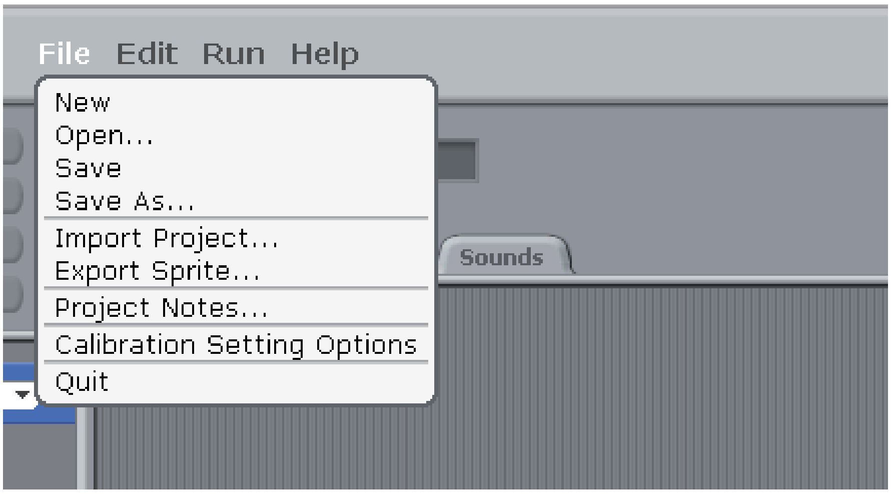
■ Variables Palette Blocks

The Variables Palette allows you to create variables and lists. You can find out more about creating variables and lists in **6.2.2.1. Variable Palette Blocks**.

6.3.5. The Main Menu

■ **The File Menu**

You can use the File menu to save and load projects.

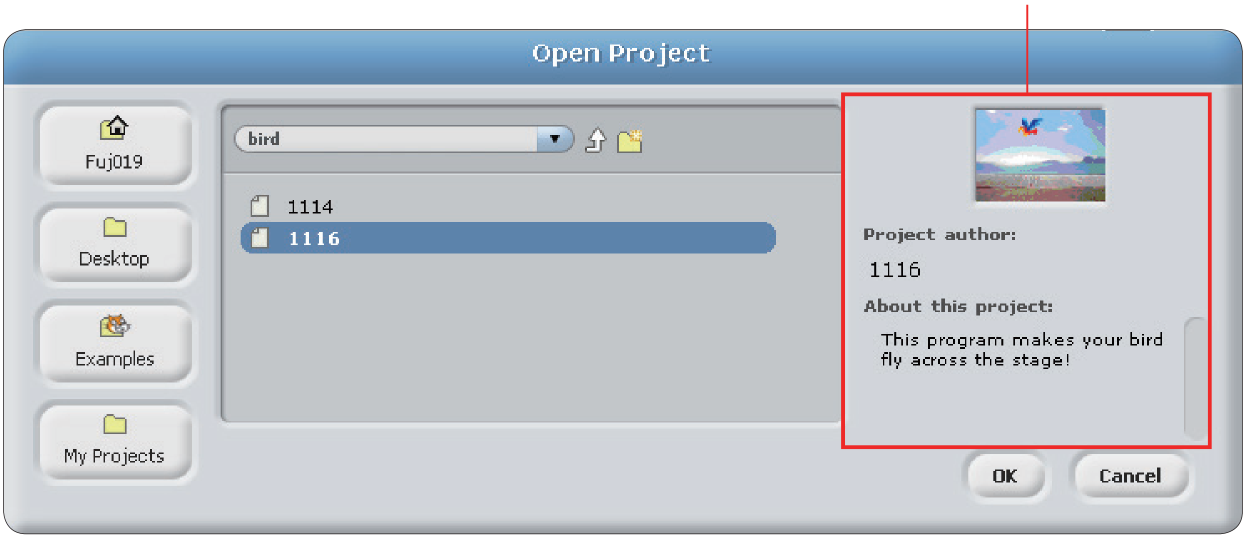


- **New**

Start a new project.

- **Open...**

Use the Open Project window to choose a Project to open. You'll see the screenshot, name, and author of any Project you select in this window.



View the details of each Project

Project author:

1116

About this project:

This program makes your bird fly across the stage!

You can also open **.bpd** files created in the Block Programming Environment and **.sb** files created in Scratch 1.4 in addition to Projects created in this software.

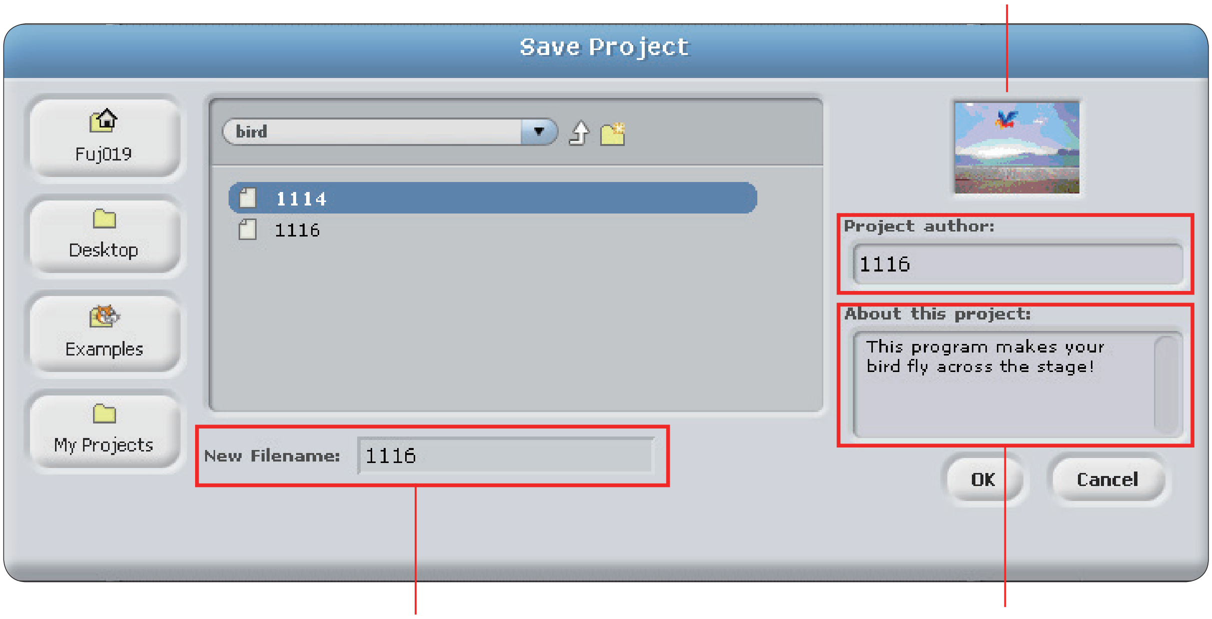
- **Save**

Saves your current project. You'll see the **Save Project** window when saving a Project for the first time.

- **Save As...**

Saves the program with a specified name. This will open the **Save Project** window. Name your Project using the **New Filename** box. You can also name the author of the Project as well as give it a brief description.

Type the author's name here



Project author: 1116

About this project: This program makes your bird fly across the stage!

New Filename: 1116

Name your Project here

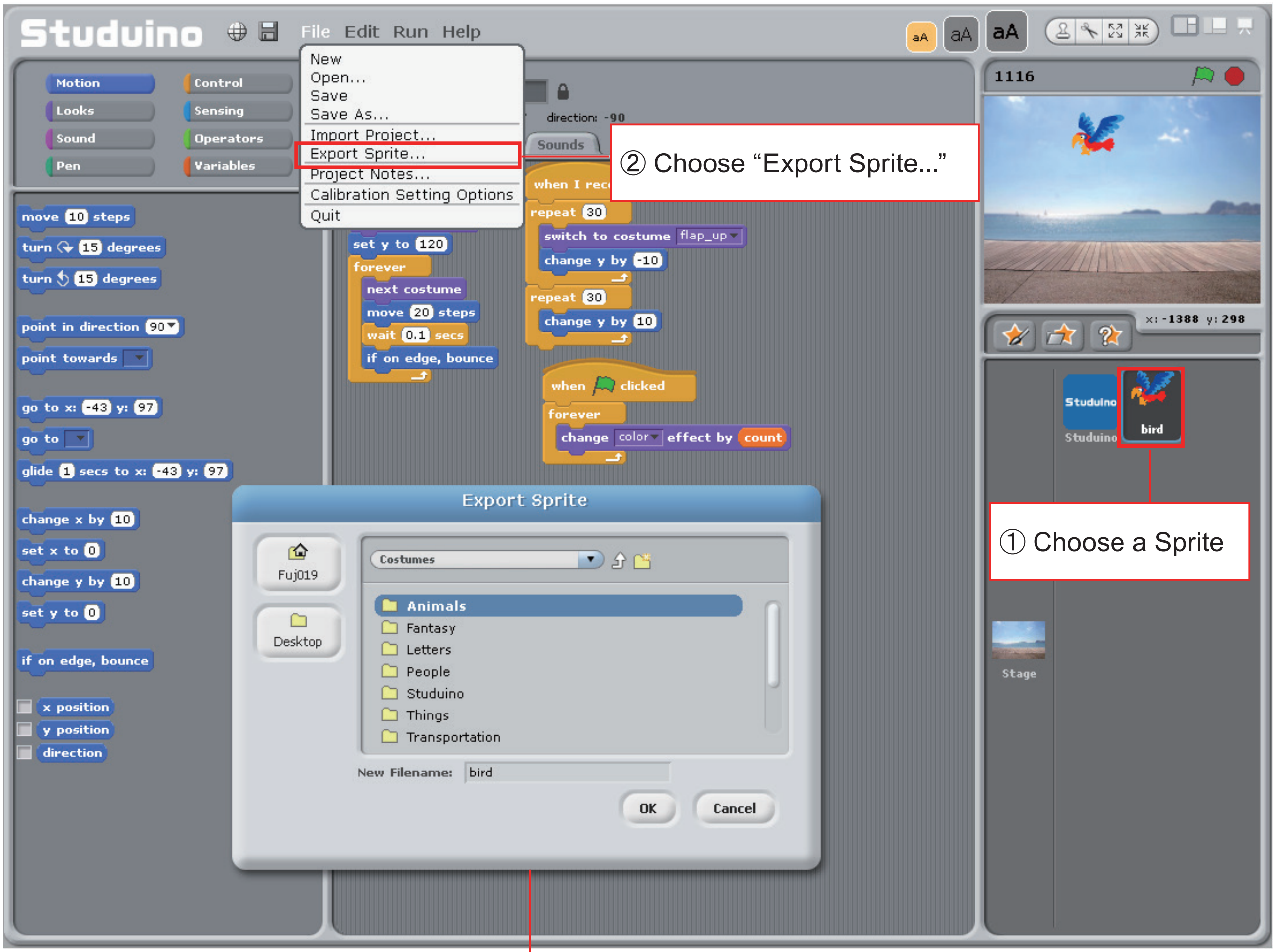
Give your Project a description here

- **Import Project...**

Load the scripts from a previous Project file. See **The File Menu** in **6.2.5. The Main Menu** for details. You can also load **.bpd** files created in the Block Programming Environment and **.sb** files created in Scratch 1.4 in addition to scripts created in this software.

- **Export Sprite...**

Exports a Sprite of your choosing. Choose the Sprite you wish you export (①). Now choose **Export Sprite...** (②) from the **File** menu and a you'll see the **Export Sprite** window (③) appear. Name your Sprite using the **New Filename** box.



New Filename: bird

② Choose "Export Sprite..."

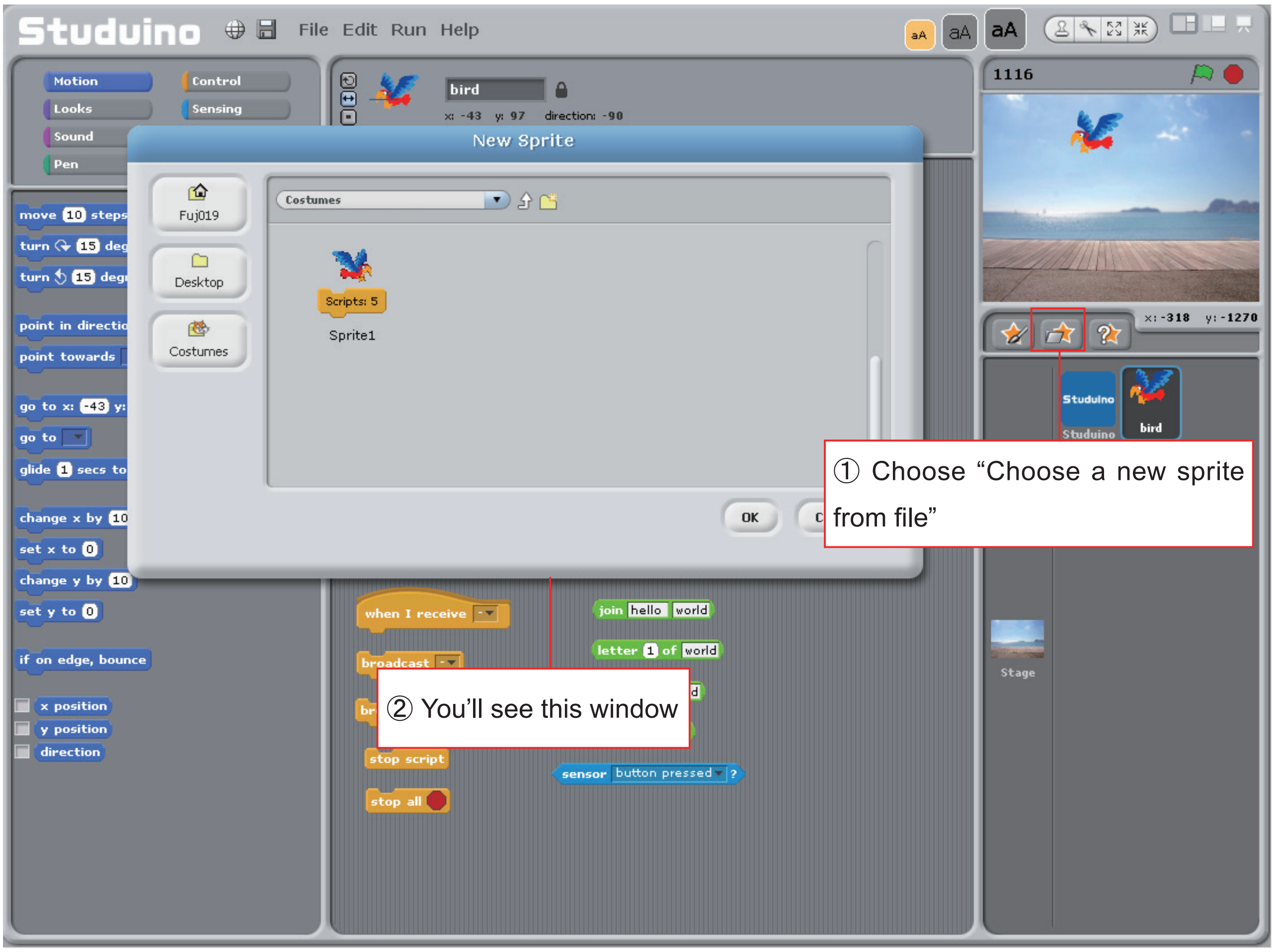
① Choose a Sprite

③ You'll see this window

You can export both regular and Studuino Sprites. Any Studuino Sprite you export will be saved with the extension **.studuinosprite**. Any regular Sprite you export will be saved with the extension **.sprite**.

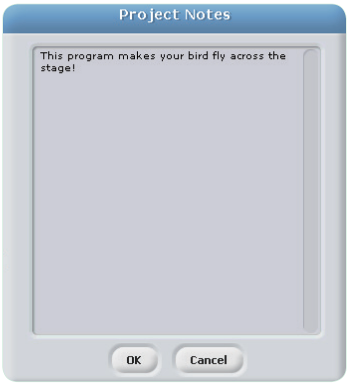
You can load any Sprite you've exported by using the **Choose a new sprite from file** button. You can also load any Sprite you've made in Scratch.

Load a Studuino Sprite and the program for that Sprite will be added to the Studuino Sprite already present in your Project.



① Choose "Choose a new sprite from file"

② You'll see this window



This program makes your bird fly across the stage!

- **Project Notes...**

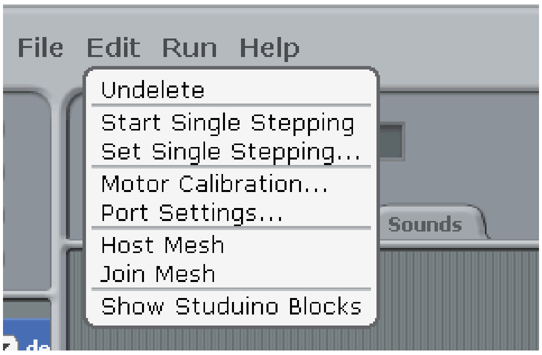
  Add or edit notes for your Project. You can do this in the Project Notes window shown to the right. Whatever you type here will be shown in the **About this project** box in the Save Project window.
- **Calibration Setting Options**

  Choose where to save your Motor Calibration settings. See The File Menu in **6.2.5. The Main Menu** for details.
- **Exit**

  Close the software.

■ The Edit Menu

The Edit menu can be used to modify your program.



- **Undelete**

  Restore a deleted block.
- **Start/Stop Single Stepping**

  Start or stop single stepping for your program. See The Edit Menu in **6.2.5. The Main Menu** for details.
- **Set Single Stepping…**

  Change how fast steps are processed. See **The Edit Menu** in **6.2.5. The Main Menu** for details.
- **Motor Calibration...**

  Use this feature to adjust the angles of your Servomotors or the speed of your DC Motors and save the settings to a file. See **The Edit Menu** in **6.2.5. The Main Menu** for details.
- **Port Settings**

  Port Settings allows you to set the parts connected to your Studuino in the Studuino Icon Programming Environment. See **The Edit Menu** in **6.2.5. The Main Menu** for details.
- **Host Mesh**

  Start hosting a network.
- **Join Mesh**

  Connect to a network on another PC.
- **Show IP Address**

  You'll only see this when connected to a network.

  It shows your PC's IP address.

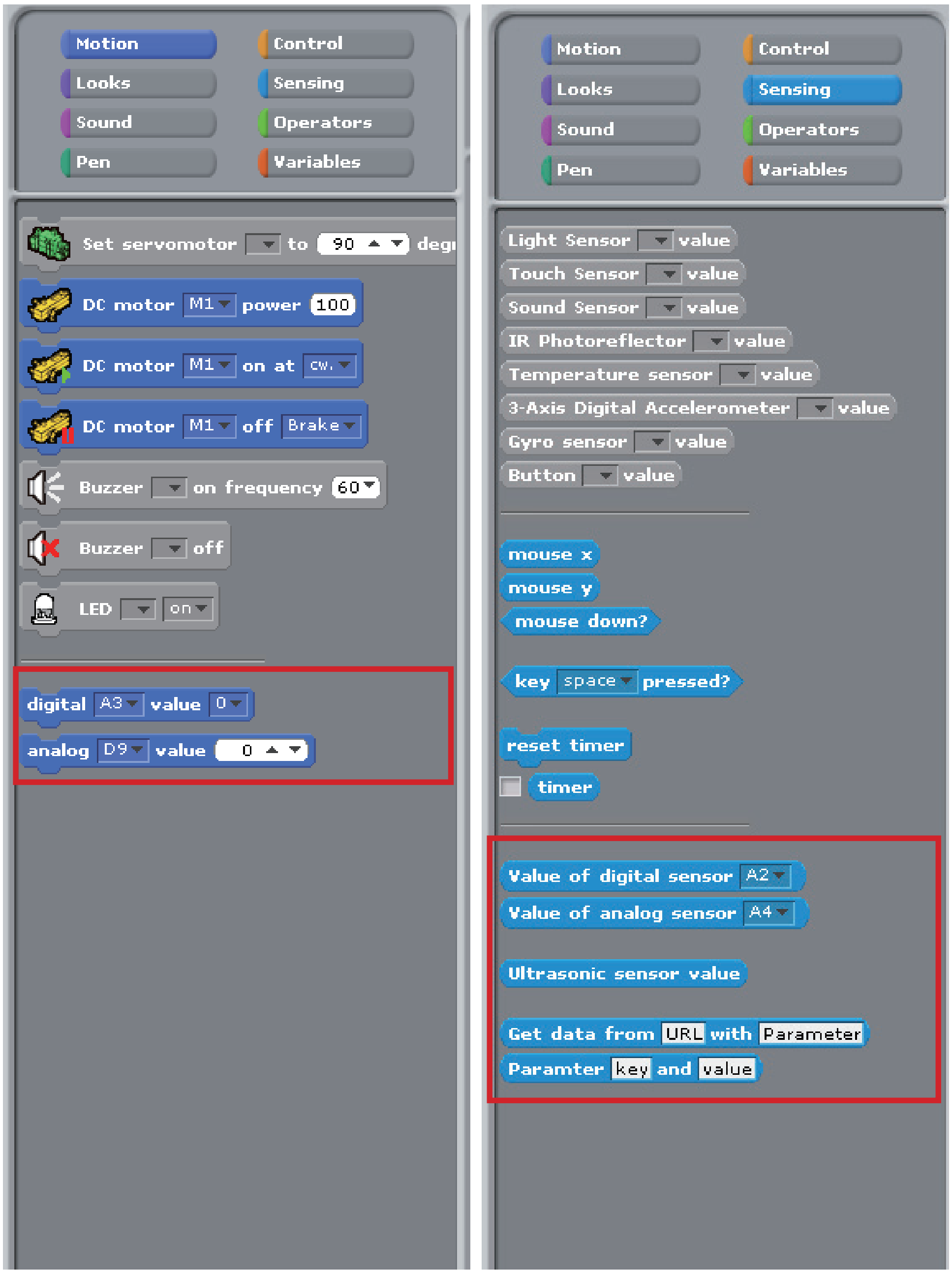
- **Stop Hosting Mesh**

  You'll only see this when connected to a network.

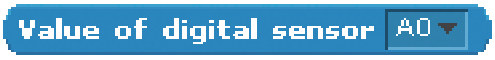
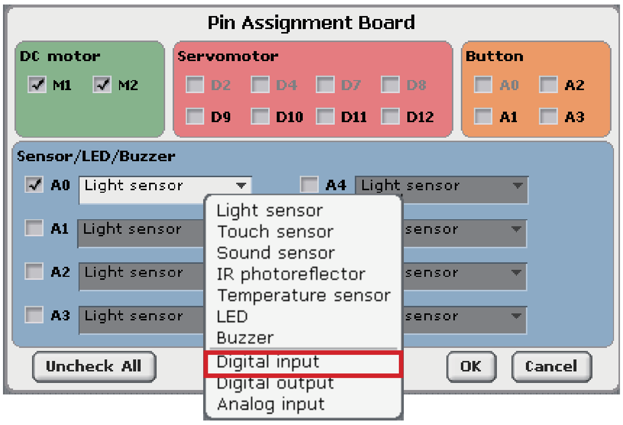
  It disconnects your PC from the network.

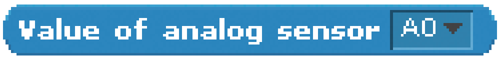
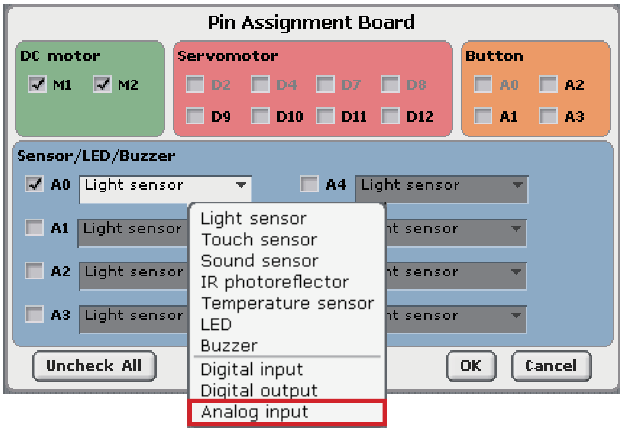
- **Show/Hide Studuino Blocks**

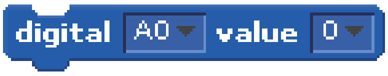
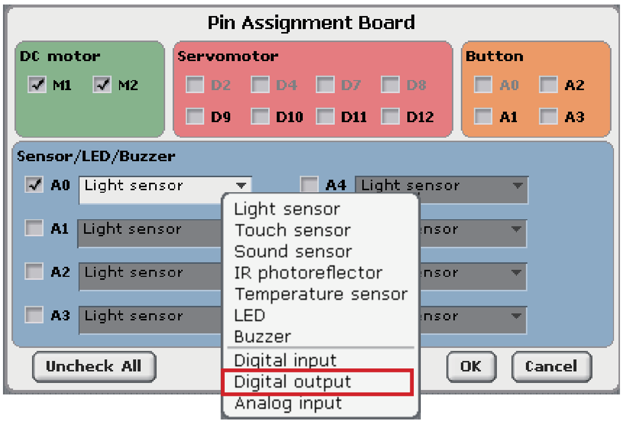
  Show or hide blocks for Servomotors, DC Motors, Buzzers, and LEDs in the Sprite Motion Palette.

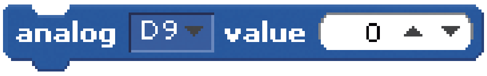
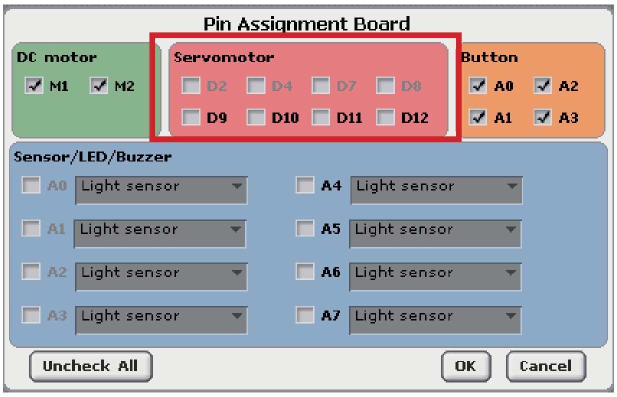


General Output Blocks

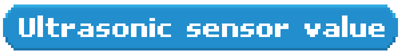
<table>
<tr><td>Block</td><td> </td></tr>
<tr><td>Overview</td><td>Looks up a digital signal.</td></tr>
<tr><td>Details</td><td>This block uses any Studuino connector from A0 to A5 as a digital input pin to find the value of the connected part.These values are either 0 or 1. When using a general digital input block, open Port Settings and choose <b>Digital input</b> for port A0-A5 as shown below.
 </td></tr>
</table>

<table>
<tr><td>Block</td><td> </td></tr>
<tr><td>Overview</td><td>Looks up an analog signal.</td></tr>
<tr><td>Details</td><td>This block uses any Studuino connector from A0 to A7 as an analog input pin to find the value of the connected part.These values are from 0 to 1023. When using a general analog input block, open Port Settings and choose <b>Analog input</b> for port A0-A7 as shown below.
 </td></tr>
</table>

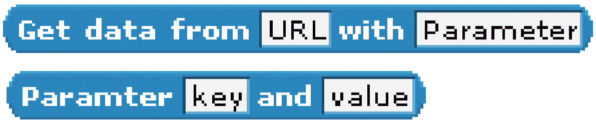
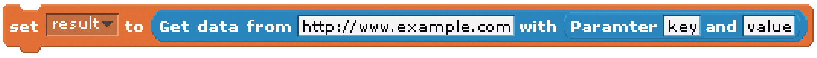
Block	
Overview	Outputs a digital signal.
Details	This block uses any Studuino connector from A0 to A5 as a digital output pin to set a value of 0 or 1 for the connected part. When using a general digital output block, open Port Settings and choose **Digital output** for port A0-A5 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button □ A0 □ A2 □ A1 □ A3 Sensor/LED/Buzzer ☑ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor Light sensor Touch sensor Sound sensor IR photoreflector Temperature sensor LED Buzzer Digital input Digital output Analog input Uncheck All OK Cancel

Block	
Overview	Outputs an analog signal.
Details	This block uses any Studuino connector from D9 to D11 as an analog output pin to set a value of 0 to 255 for the connected part. When using a general analog output block, open Port Settings and uncheck Servomotor boxes D2 to D12 as shown below.  Pin Assignment Board DC motor ☑ M1 ☑ M2 Servomotor □ D2 □ D4 □ D7 □ D8 □ D9 □ D10 □ D11 □ D12 Button ☑ A0 ☑ A2 ☑ A1 ☑ A3 Sensor/LED/Buzzer □ A0 Light sensor □ A1 Light sensor □ A2 Light sensor □ A3 Light sensor □ A4 Light sensor □ A5 Light sensor □ A6 Light sensor □ A7 Light sensor Uncheck All OK Cancel

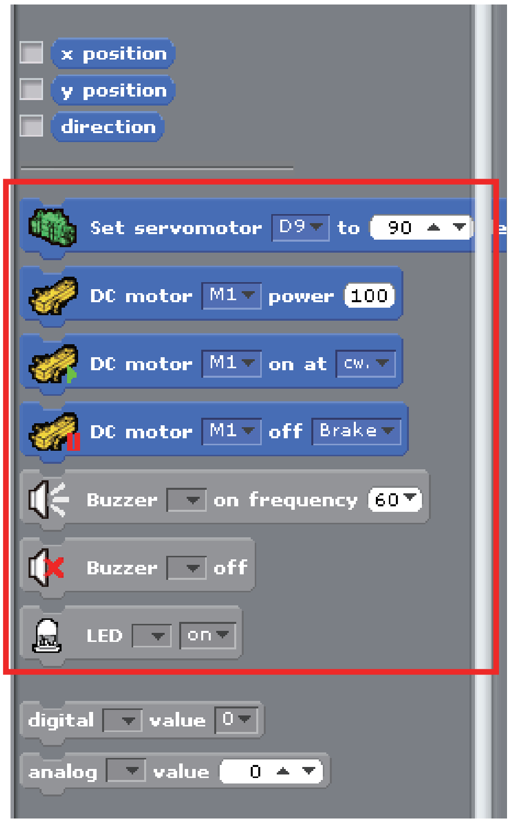
Ultrasonic Sensor Block

Block	 Ultrasonic sensor value
Overview	Looks up the value of the Ultrasonic Sensor.
Details	**Block Programming Environment Guide (Optional Parts)** to learn how to use this block. http://www.artec-kk.co.jp/studuino/docs/en/Studuino_tutorial_block_3.pdf

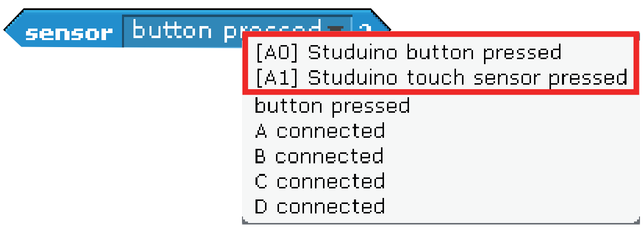
HTTP Block

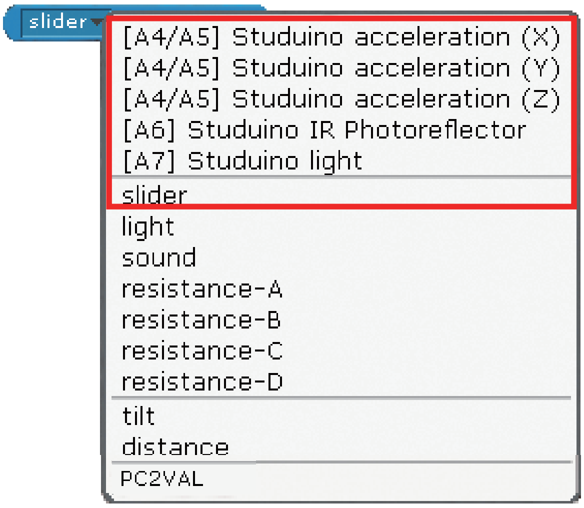
Block	 Get data from URL with Parameter Paramter key and value
Overview	Connects to the internet and looks up information on a server.
Details	This block can receive information from internet servers. Your computer must be connected to the internet to use this block. To use this block to access the internet, you must know the specifications and procedure for accessing the server you're connecting to. Ex)  set result to Get data from http://www.example.com with Paramter key and value The block shown here executes the following commands. 1) Connect to http://www.example/com?key=value 2) Save the information obtained from the server in the variable **result**

You can also add Extra Blocks to the Stage and sprites to get access to blocks that control your ArtecRobo sensors and actuators, like these:



As shown below, you can also use slider sensor value and sensor button pressed ? blocks to retrieve the values of the sensors connected to your Studuino. (For example, let's say the Port Settings have **A0** set to the **Studuino button**, **A1** to a **Touch Sensor**, **A4/A5** to an **Accelerometer**, **A6** to an **IR Photoreflector**, and **A7** to a **Light Sensor**. In this case, you can select which of those sensor values you want to use for the slider sensor value and sensor button pressed ? blocks through the menus shown below.)




slider
[A4/A5] Studuino acceleration (X)
[A4/A5] Studuino acceleration (Y)
[A4/A5] Studuino acceleration (Z)
[A6] Studuino IR Photoreflector
[A7] Studuino light
slider
light
sound
resistance-A
resistance-B
resistance-C
resistance-D
tilt
distance
PC2VAL

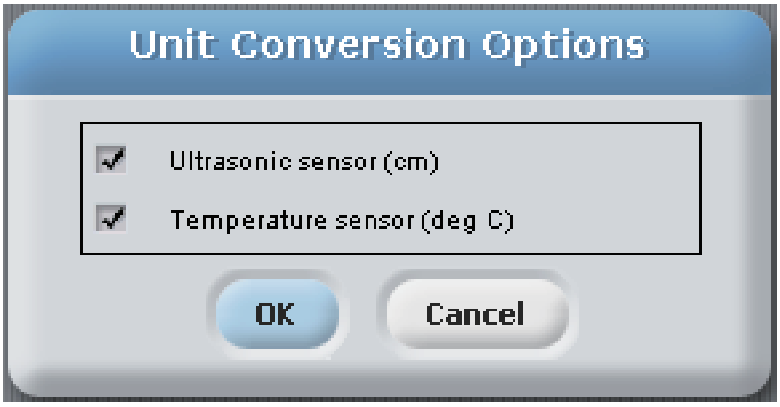
- **Hide Extra Blocks**

  This item is only available in the menu once you've chosen Show **Extra Blocks**.

  It hides the extra blocks.

- **Sensor Value Settings**

This lets you set what values a sensor block looks up. Select the main heading, and you'll see the dialog box shown below.



Unit Conversion Options

☑ Ultrasonic sensor (cm)

☑ Temperature sensor (deg C)

By toggling the check boxes in this dialog box on/off, you can change the units used in a sensor block's valus as follows.

	Check	
Sensor Type	On	Off
Ultrasonic Sensor	Distance (cm)	Time (microseconds)
Temparature Sensor	Temperature Sensor Temperature [℃]	Numbers

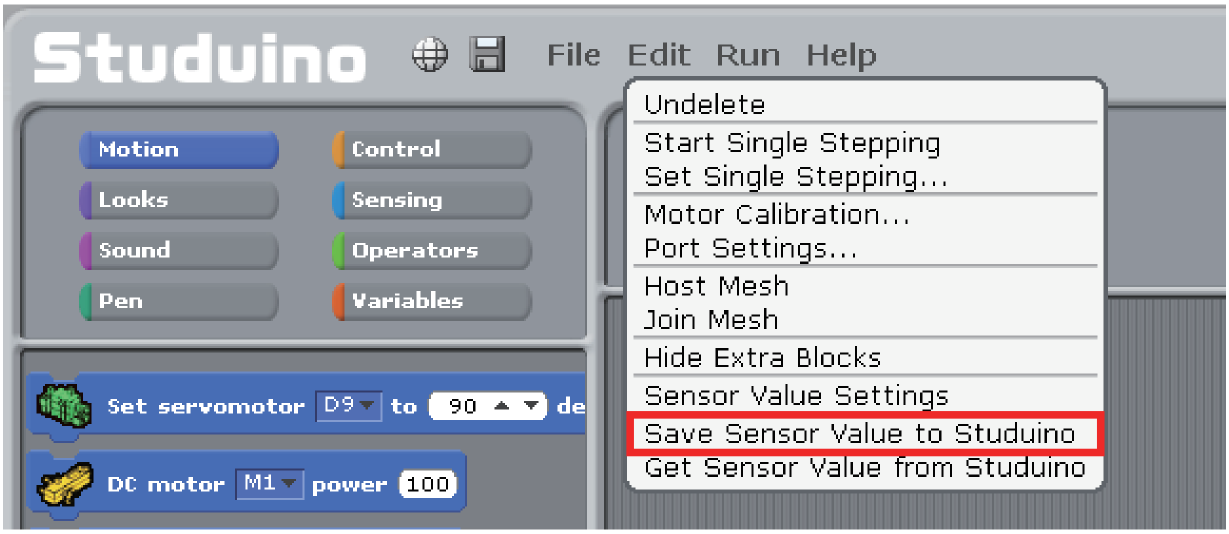
By removing the check mark for the Ultrasonic Sensor in this dialog box, you can make your Ultrasonic Sensor measure the time it takes the Sensor's sound waves to bounce back after they are released. Values range from 0 to 30,000 microseconds. By removing the check mark for the Temperature Sensor in this dialog box, you can read the sensor's values directly. Values range from -32768 to 32767.

- Recording and Retrieving Sensor Values

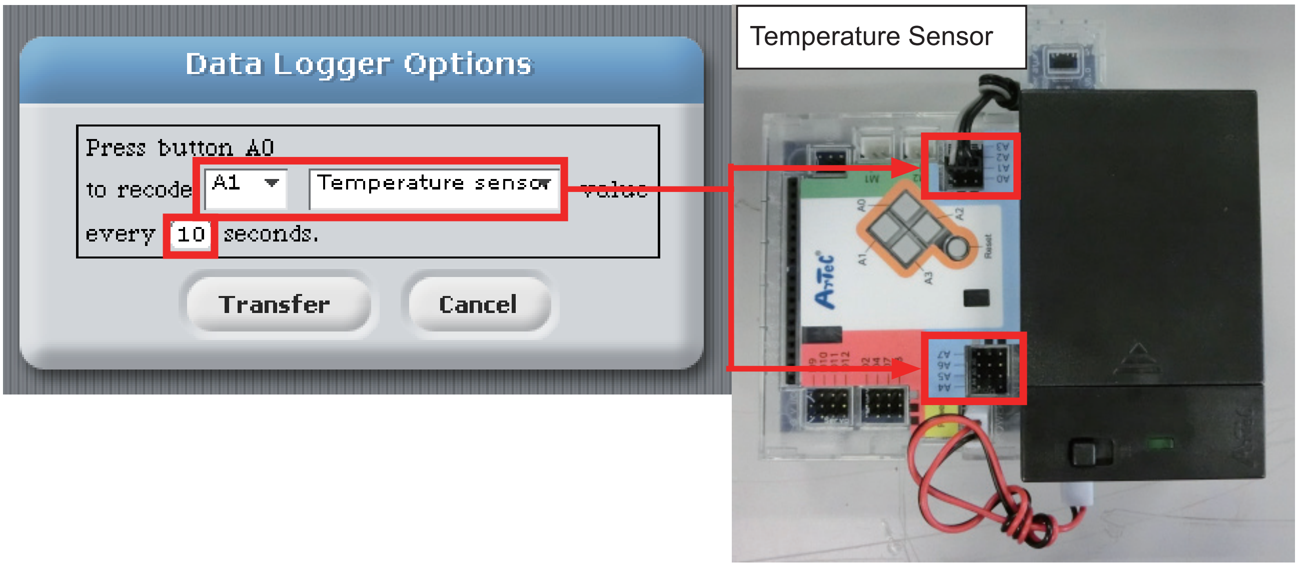
You can use your Studuino to record values from your sensors. In the following section, you'll learn how to use the menus to save sensor values to your Studuino and how to retrieve values you've recorded there.

■ Recording Sensor Values

1. Open the **Edit** menu and select **Save Sensor Value to Studuino**.



2. This opens the **Data Logger Options** dialog box, where you can set what type of sensor part you're recording from, what port it's connected to, and the interval for recording data from it.


Data Logger Options

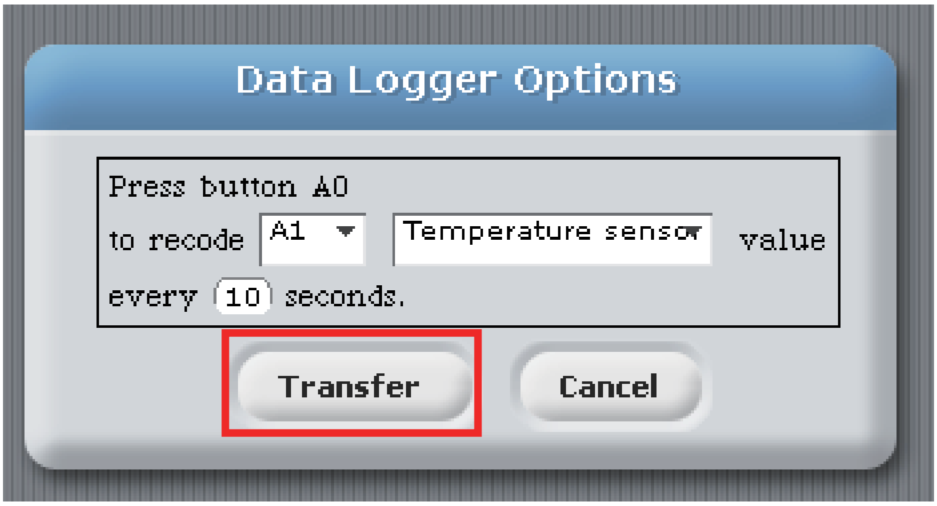
Press button A0

to recode A1 Temperature sensor value

every 10 seconds.

Port A0 is used to start recording your sensor values, which leaves ports A1 to A7 available for your sensors. The recording interval should be no less than one millisecond (0.001 seconds) and no more than one day (86400 seconds). While your Studuino can record up to 250 items of data, it can only record for up to 50 days continuously. This means that a recording interval set to once daily will only record 50 values (50 days' worth of data).

3. Use a USB cable to connect your Studuino to your PC, then click the **Transfer** button in the dialog box to transfer the recording program to your Studuino.



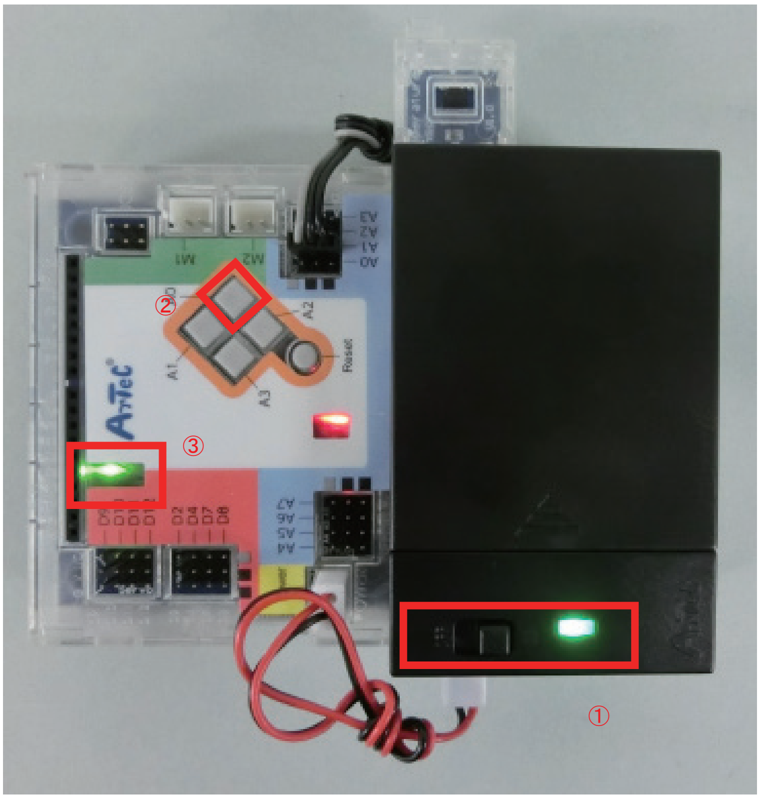
Data Logger Options

Press button A0

to recode A1 Temperature sensor value

every 10 seconds.

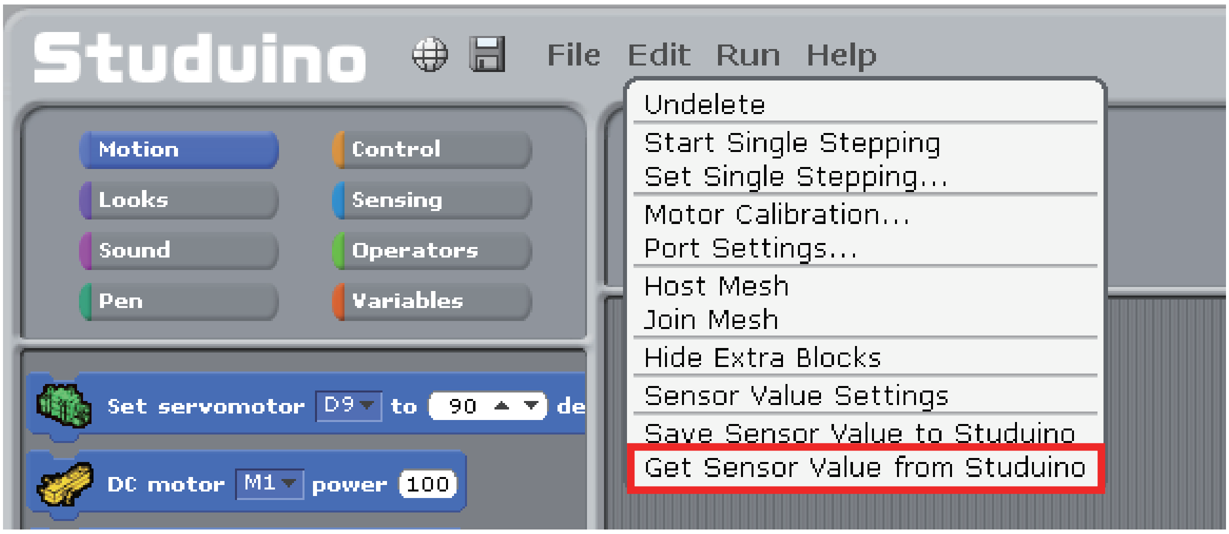
4. Once the transfer is complete, remove your Studuino, turn your Battery Box on, then press button A0 on the Studuino to start recording sensor values. The green LED light on the Studuino will blink once every second while the recording is in progress.



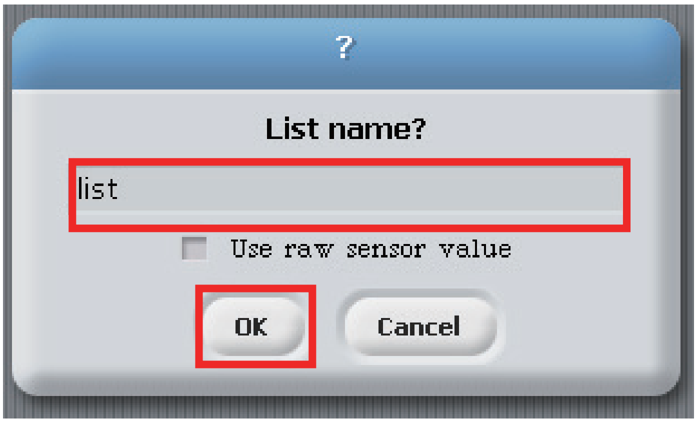
①	Battery Box ON
②	Press A0
③	The LED blinks

■ Retrieving Sensor Values

1. When your Studuino has recorded sensor values, connect it to your PC with a USB cable, then choose **Get Sensor Value** from Studuino from the **Edit** menu.

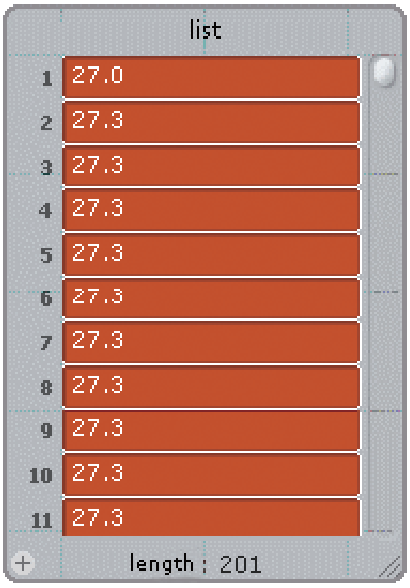


2. Use the dialog box that appears to name your list and click **OK**.



By default the **Use raw sensor value** box is unchecked. This setting will output a list using the Studuino values for a given sensor (e.g. 1-100 for a Light Sensor, degrees for a Temperature sensor). If you check the **Use raw sensor value** box, the values will be listed in their original format (numbers 0-1023).

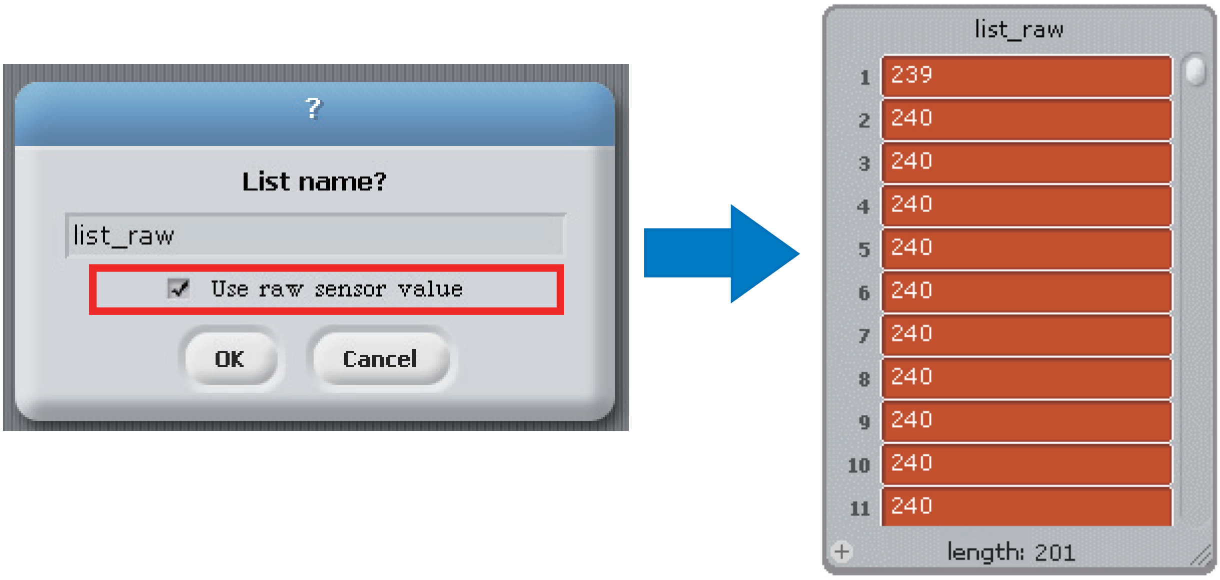
Example (Temperature Sensor)



```
list
1  27.0
2  27.3
3  27.3
4  27.3
5  27.3
6  27.3
7  27.3
8  27.3
9  27.3
10 27.3
11 27.3
length : 201
```

In the default settings (with **Use raw sensor value** unchecked), the values from the Temperature Sensor are translated into degrees before being stored in a list.

With **Use raw sensor value** checked, the sensor values are not translated. Instead they are added to the list in their original format.



?

List name?

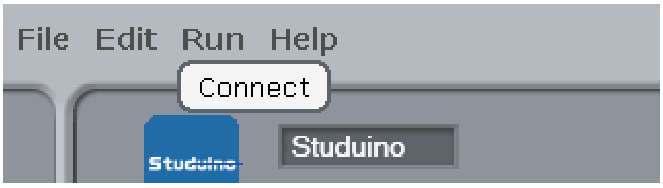
list_raw

☑ Use raw sensor value

```
list_raw
1  239
2  240
3  240
4  240
5  240
6  240
7  240
8  240
9  240
10 240
11 240
length: 201
```

■ **The Run Menu**

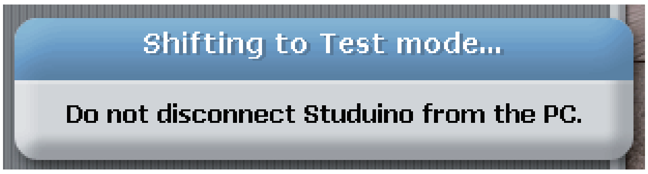
The Run menu allows you to link with your Studuino while making programs.



• **Connect/Disconnect**

Use this to communicate with your Studuino and control parts in real time. Turn this feature on or off by choosing **Connect** or **Disconnect** from the **Run** menu. Make sure your PC and Studuino are connected before choosing Connect.

You'll see the message below when connecting.



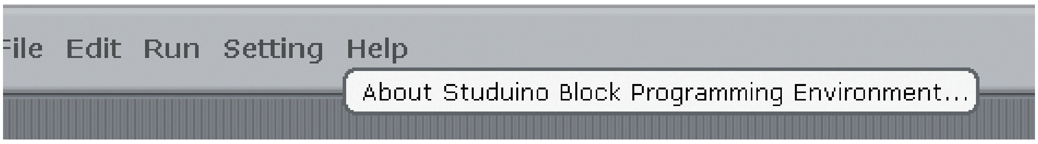
Shifting to Test mode...

Do not disconnect Studuino from the PC.

Do not disconnect your Studuino from the PC when connecting.

■ **Help Menu**

The Help menu contains information about the Studuino Block Programming Environment.



■ About Studuino Block Programming Environment…

Shows information about the Studuino Block Programming Environment.

■ **FAQ**

See the FAQ section of the Artec website for help.

6.3.6. The Context Menu

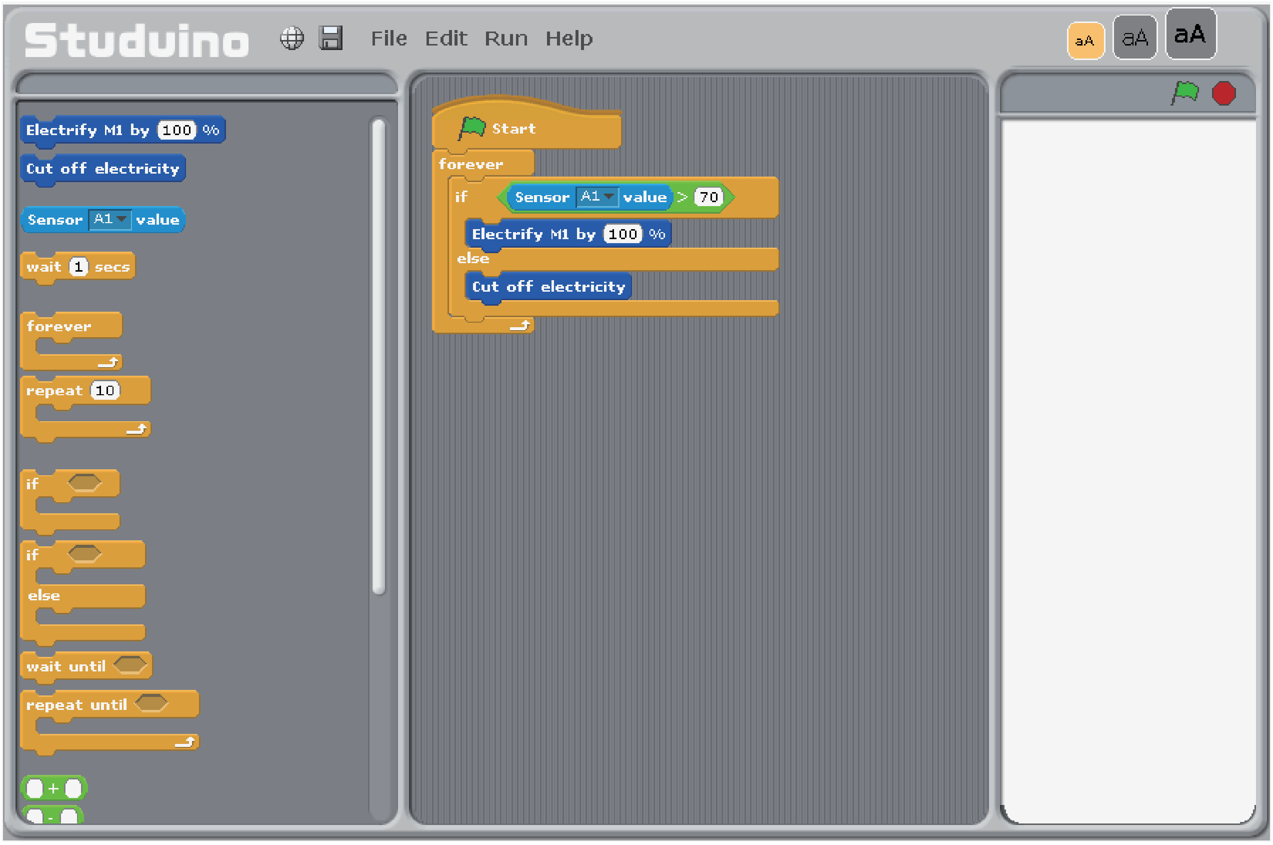
See **6.2.6. The Context Menu** for details.

6.3.7. For Windows Tablets

See **6.2.7.** for more details.

6.4. BPE Experiments

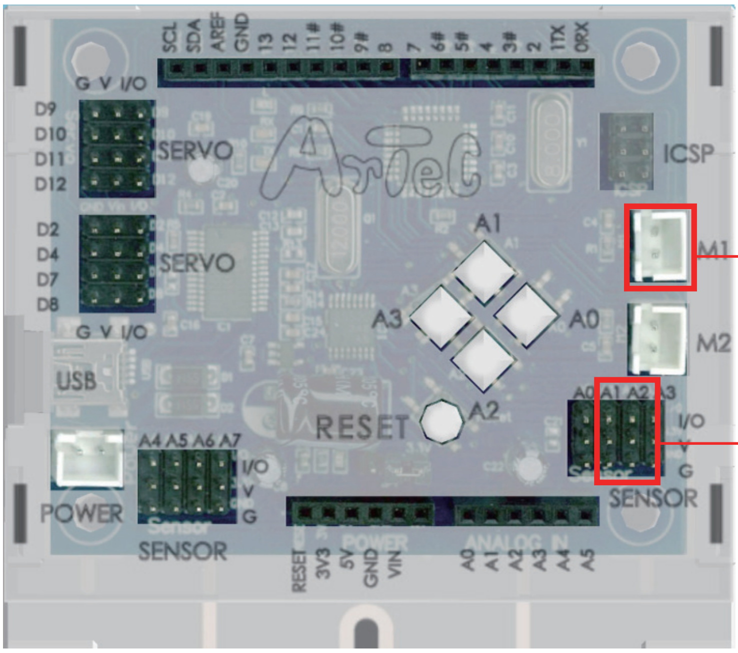
6.4.1. Overview and Features



```
Start
forever
    if Sensor A1 value > 70
        Electrify M1 by 100 %
    else
        Cut off electricity
```

The Experiments build of the Studuino Block Programming Environment allows you to connect a test lead (product #151097, Alligator Test Lead for Studuino) to M1 on your Studuino and use your sensor values to change the amount of electricity running to devices like commercial miniature light bulbs, motors, Buzzers, or LEDs.

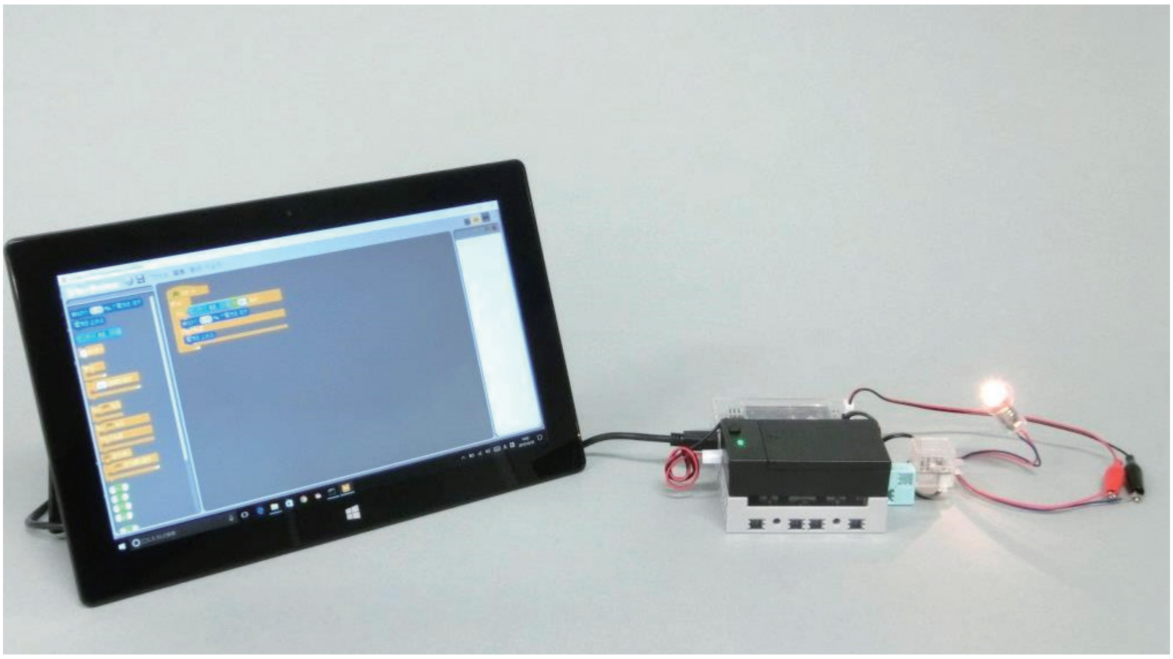
This version of the software only uses Studuino connectors A1, A2, and M1.



Connect the electronic device here

Connect any type of sensor here

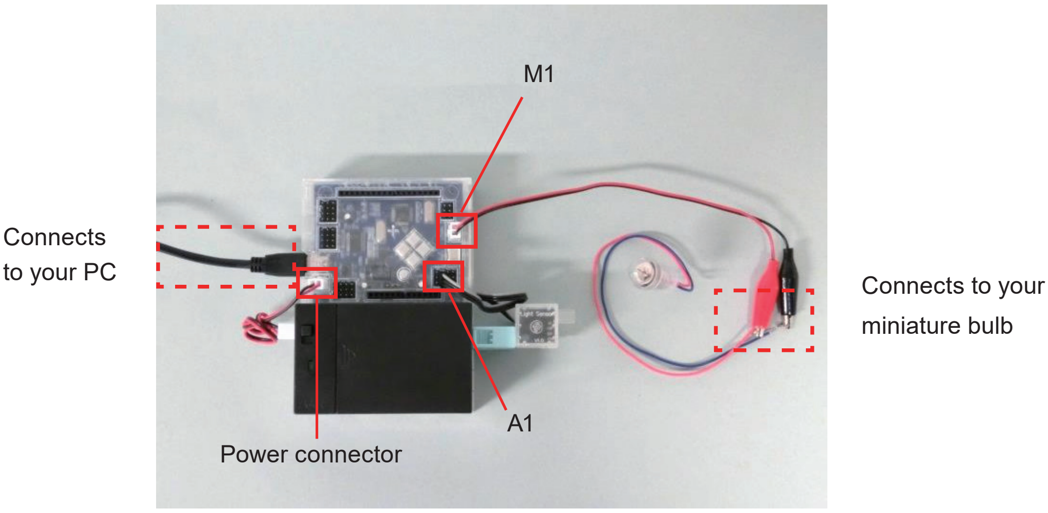
6.4.2. Programming



Here we'll program a Sensor Light that turns a miniature light bulb on and off using a Light Sensor connected to your Studuino.

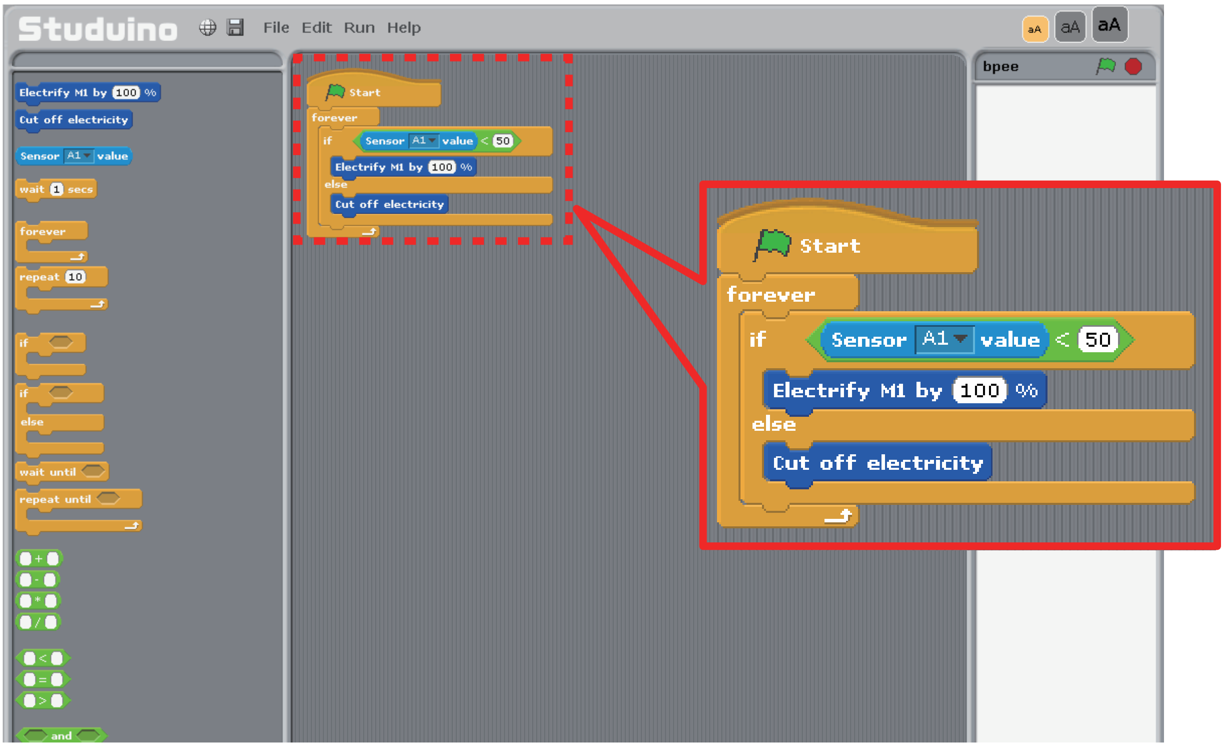
6.4.2.1. Studuino Settings

Plug your **Light Sensor** into **A1** on your Studuino. Plug your **test lead** into **M1**, connect it to your miniature bulb, and plug your **Battery Box** into the **Power** connector. Now connect your Studuino to the PC using a USB cable.



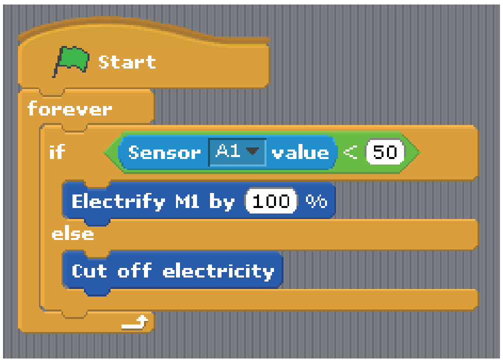
6.4.2.2. Programming

We’re going to make the program shown below in the Script Field.



```
Start
forever
  if Sensor A1 value < 50
    Electrify M1 by 100 %
  else
    Cut off electricity
```

The program we’ll make will do the following.

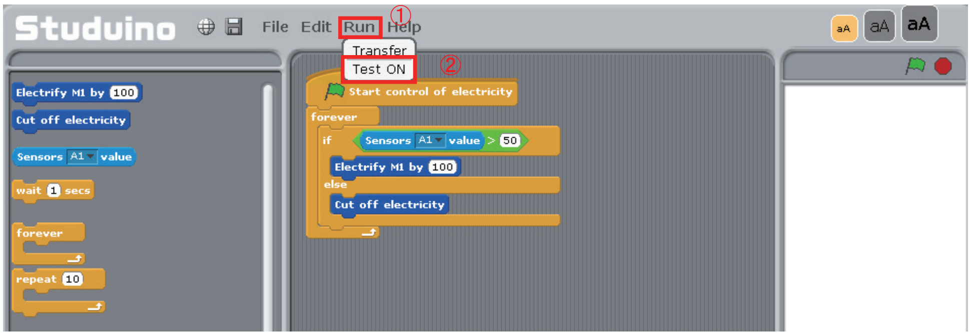


```
Start
forever
  if Sensor A1 value < 50
    Electrify M1 by 100 %
  else
    Cut off electricity
```

6.4.2.3. Checking Your Program in Test Mode

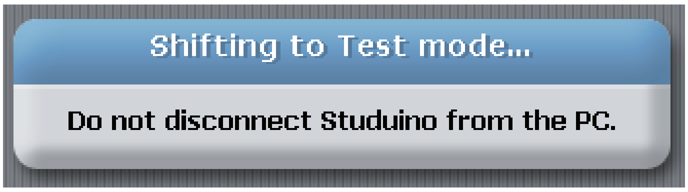
You can use Test Mode to see how your program works in real time.

Click the **Run** menu (①) and choose **Test ON** (②) to start Test Mode.



```
Start control of electricity
forever
  if Sensors A1 value > 50
    Electrify M1 by 100
  else
    Cut off electricity
```

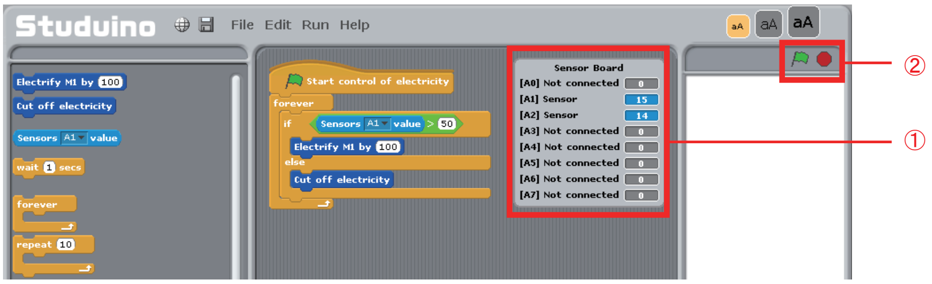
You'll see the message below when Test Mode is opening.



Shifting to Test mode...

Do not disconnect Studuino from the PC.

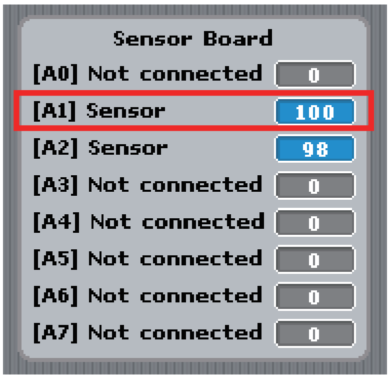
The message will disappear and the **Sensor Board** (①) will appear. Click the **green flag** (②) and turn your Battery Box on. The blocks currently running will be highlighted in yellow.



```
Start control of electricity
forever
  if Sensors A1 value > 50
    Electrify M1 by 100
  else
    Cut off electricity
```

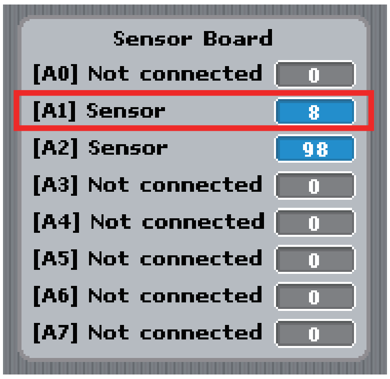
```
Sensor Board
[A0] Not connected   0
[A1] Sensor         15
[A2] Sensor         14
[A3] Not connected   0
[A4] Not connected   0
[A5] Not connected   0
[A6] Not connected   0
[A7] Not connected   0
```

The Sensor Board shows the values of any sensor connected to your Studuino. Try covering the Light Sensor on A1 with your hand to block out the light. The value of **[A1] Sensor** should become smaller.



```
Sensor Board
[A0] Not connected   0
[A1] Sensor        100
[A2] Sensor         98
[A3] Not connected   0
[A4] Not connected   0
[A5] Not connected   0
[A6] Not connected   0
[A7] Not connected   0
```

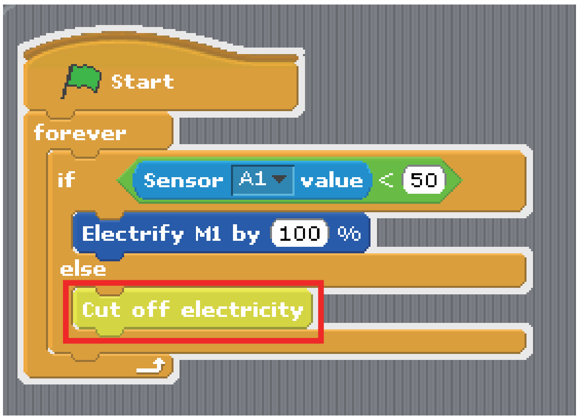
In a well-lit room



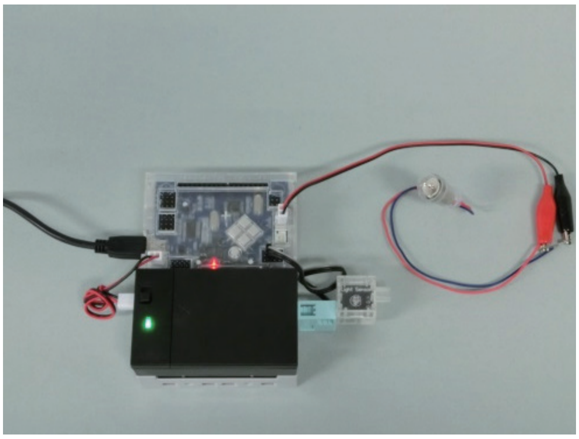
```
Sensor Board
[A0] Not connected   0
[A1] Sensor          8
[A2] Sensor         98
[A3] Not connected   0
[A4] Not connected   0
[A5] Not connected   0
[A6] Not connected   0
[A7] Not connected   0
```

Shaded by a hand

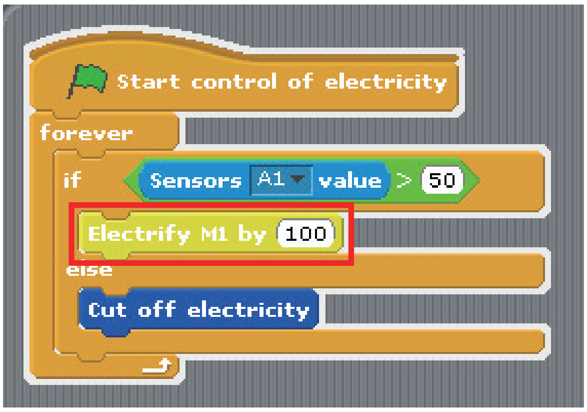
The program you're running in Test Mode will run the Cut off electricity block to cut off the electricity to M1 and turn off the miniature bulb when the Light Sensor value is over 50 in a well-lit room.



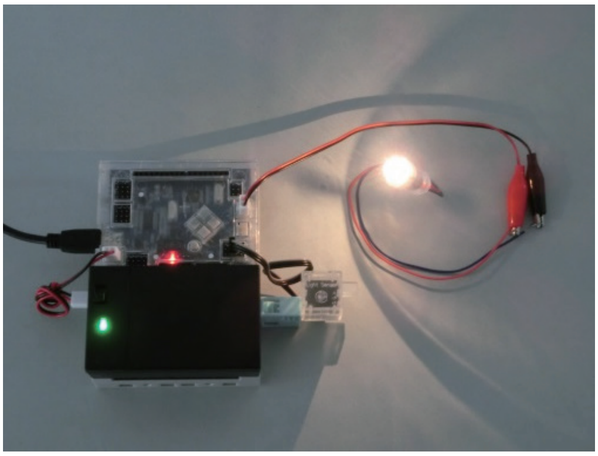
```
Start
forever
  if Sensor A1 value < 50
    Electrify M1 by 100 %
  else
    Cut off electricity
```



When the Light Sensor value is under 50 in a dimly lit room, this program will run the Electrify M1 by 100 % block to send electricity to M1 and turn on the miniature bulb.



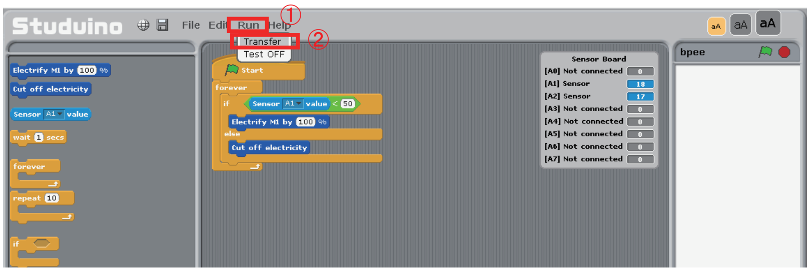
```
Start control of electricity
forever
  if Sensors A1 value > 50
    Electrify M1 by 100
  else
    Cut off electricity
```



6.4.2.4. Transferring a Program

Once you're sure that the program works, you can Transfer it to your Studuino and run it without the need for a USB cable.

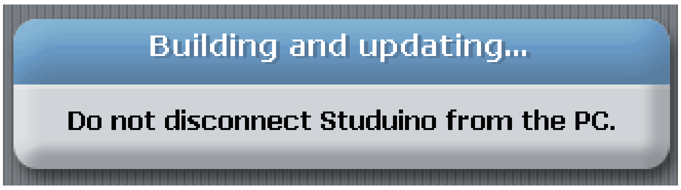
Turn your Battery Box off. Now click the **Run** menu (①) and choose **Transfer** (②).



```
Start
forever
  if Sensor A1 value < 50
    Electrify M1 by 100 %
  else
    Cut off electricity
```

Sensor Board	
[A0] Not connected	0
[A1] Sensor	18
[A2] Sensor	17
[A3] Not connected	0
[A4] Not connected	0
[A5] Not connected	0
[A6] Not connected	0
[A7] Not connected	0

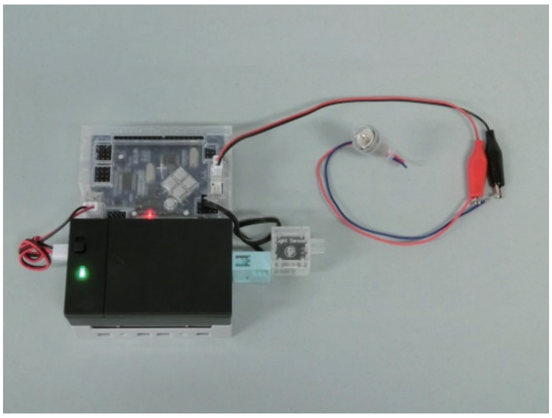
You'll see the message below when transferring your program.



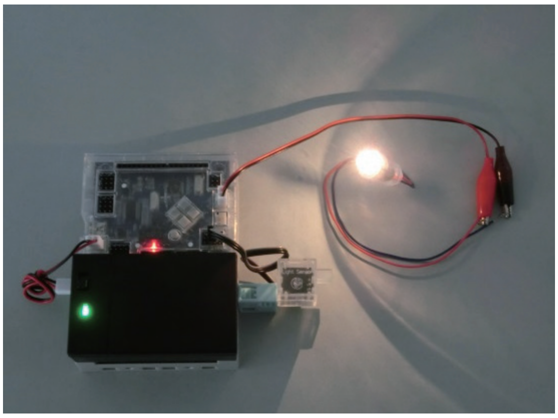
Building and updating...

Do not disconnect Studuino from the PC.

Once the message disappears, unplug the USB cable from your Studuino and turn the Battery Box on. Just like in Test Mode, your bulb will stay off when the room is bright and turn on when the room darkens.

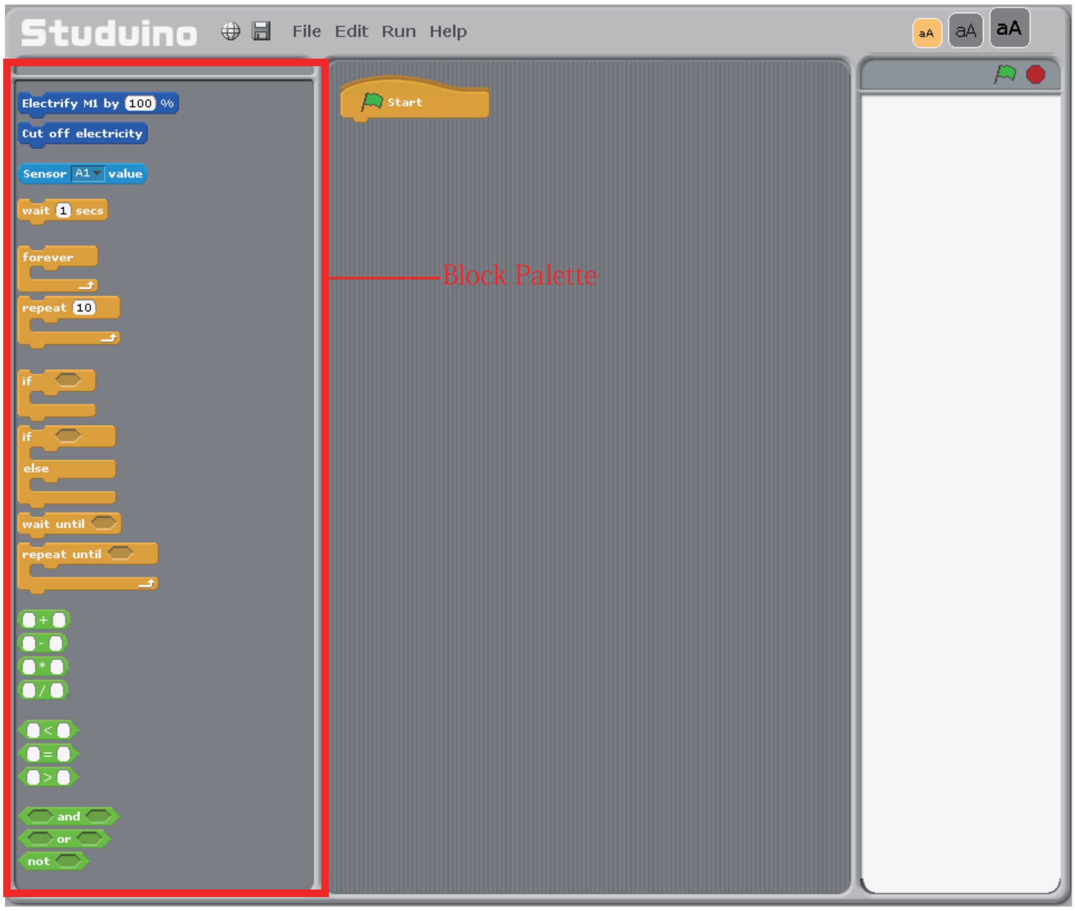


When the room is bright

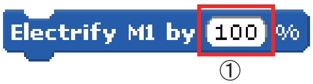
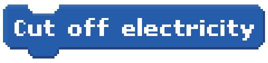
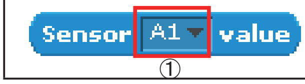


And when the room is dark

6.4.3. More About Blocks



This version of the software keeps all of the blocks you’ll need to use in a single Palette. Follow along below to learn about the blocks that are exclusive to this version. See **6.2.2.1. More About Blocks** for details on the other blocks.

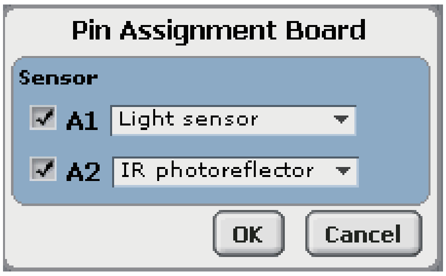
Block	Description
	Sends the amount of electricity specified in ① to connector M1.
	Cuts off the electricity being sent to M1.
	Looks up the value of the sensor connected to ① .

6.4.4.The Main Menu

See below to learn more about the software menus. See **6.2.5. The Main Menu** for details on the other items in the Main Menu.

- **The Edit Menu**
- **Port Settings...**

  Port Settings allows you to set the sensor connected to your Studuino. Choosing Port Settings will open the Port Settings window. You can set a Light Sensor, IR Photoreflector, or Temperature Sensor.



Pin Assignment Board

Sensor

☑ A1 Light sensor

☑ A2 IR photoreflector

6.4.5. The Context Menu

See **6.2.6. The Context Menu** for more details.

6.4.6. For Windows Tablets

See **6.2.7.** for more details.

6.5. For Windows Tablets

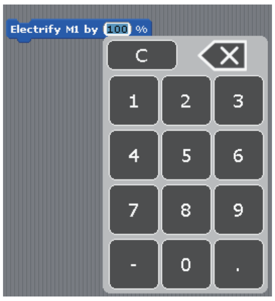
Starting with Studuino Block Programming Environment 1.4.0.0, the following features have been added or changed in order to work with Windows tablets.

Touch Keyboard Support for Windows 10 Tablet Mode

Using Windows 10 in Tablet Mode will automatically bring up the on-screen touch keyboard when input is required.

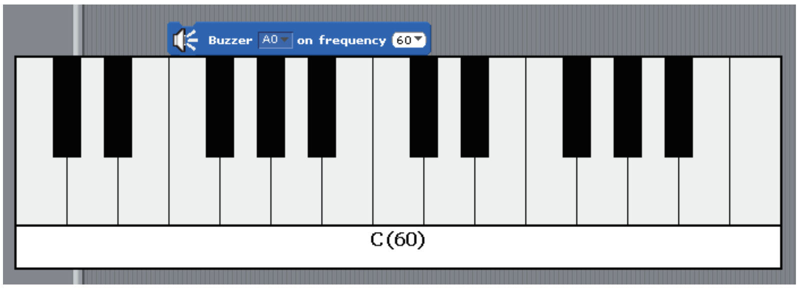
Number Pad Display for Numerical Values

The software will show a number pad instead of the touch keyboard when inputting numerical values into a block.



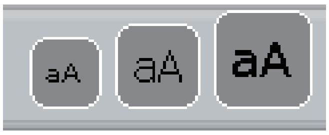
Larger Musical Keyboard

The size of the musical keyboard has been increased to allow for easier selection of Buzzer notes.



Font Sizes

The user can now choose between small, medium, and large fonts. Choosing a larger font size on a Windows tablet will also increase the size of the blocks, making it easier to touch them.



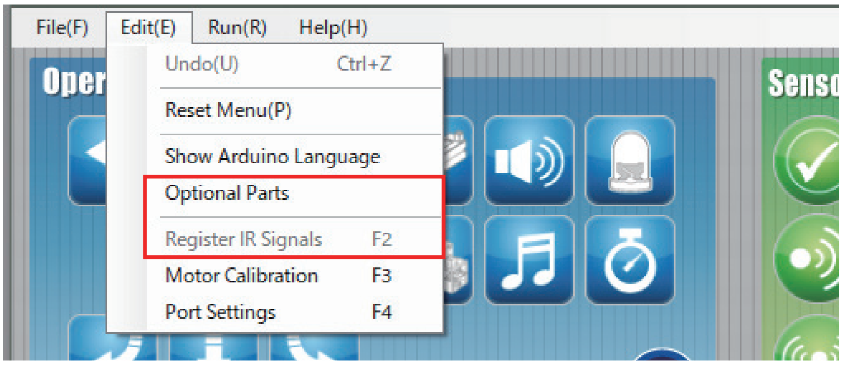
6.6. Studuino 2.0

This section describes new features found in Studuino Software ver. 2.0

6.6.1. The Icon Programming Environment

- **The Edit Menu**

The Edit Menu in ver. 2.0 now contains the additional items Optional Parts and Register IR Signal.



- **Optional Parts**

Check this option if you wish to use Gyroscopes, Color Sensors, Ultrasonic Sensors, Temperature Sensors, and Bluetooth Modules. Visit our website and download the Optional Parts manual for details on how to use these parts.

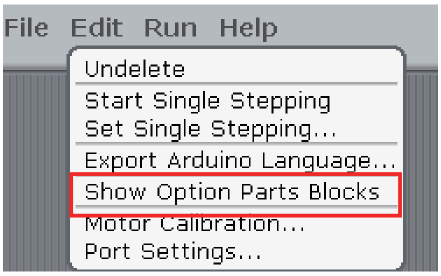
- **Register IR Signal**

This opens the registration window, allowing you to register IR signals. You'll need to enable optional parts and set your IR Receiver in Port Settings.

6.6.2. Block Programming Environment

- **The Edit Menu**

The Edit Menu in ver. 2.0 now contains the additional item Show Optional Parts.



- **Show Optional Parts**

This option shows blocks for Ultrasonic Sensors, IR Receivers, Temperature Sensors. Gyroscopes, Color Sensors, and Bluetooth in the Sensing Palette.

- **Hide Optional Parts**

You'll only see this option after selecting Show Optional Parts. Selecting this will hide the optional part blocks in the Sensing Palette.

Follow the link below to download and read the Block Programming Environment Optional Parts manual for details on these optional parts and their associated blocks.

http://www.artec-kk.co.jp/studuino/ja/studuino_v2.php

7. Troubleshooting

See the FAQ section of the Artec website for help with troubleshooting this software.

http://www.artec-kk.co.jp/studuino/en/faq.php

If you’re still having trouble or can’t find your issue on the above page, feel free to contact us directly.